DEVICE AND METHOD FOR CONTROLLING FA SYSTEM, AND CONTROL PROGRAM GENERATING METHOD

Patent number: JP9171405
Publication date: 1997-06-30
Inventor: KOYAMA MASAHIRO; MIYAKE NORIHISA
Applicant: HITACHI LTD
Classification:
- international: G05B19/418; G05B13/02; G05B15/02; G06F9/06
- european:
Application number: JP19950331666 19951220
Priority number(s): JP19950331666 19951220

View INPADOC patent family

Abstract of **JP9171405**

PROBLEM TO BE SOLVED: To describe control programs of FA system integrating into one, to automatically allocate it to respective processors and to convert the control program into programs for these processors. SOLUTION: The work specifications of respective cells consisting of the FA system are described in a single language, program module groups by functions are generated for performing sequence control, control of automatic machine and control of programmable peripheral equipment, and these respective program modules 1601 by functions are separated into program groups 1803 by processors composed of plural programs 1802 by processors. At this time, a unit block processing time estimating and evaluating means 1801d evaluates the processing time of each unit block corresponding to processor configuration and function data 1801c, and a unit block connecting means 1801e connects unit blocks 1602 so as to optimize the distribution of processing load at plural processors.

(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（Ａ）**　(11)特許出願公開番号

特開平9－171405

(43)公開日　平成９年(1997)６月30日

(51)Int.Cl.[6]	識別記号	庁内整理番号	ＦＩ		技術表示箇所
Ｇ０５Ｂ 19/418			Ｇ０５Ｂ 19/417	Ａ	
13/02			13/02	Ｊ	
15/02			Ｇ０６Ｆ 9/06	５３０Ｖ	
Ｇ０６Ｆ 9/06	５３０			５３０Ｓ	
		0360－3Ｈ	Ｇ０５Ｂ 15/02	Ｐ	

審査請求　未請求　請求項の数26　ＯＬ　（全 38 頁）

(21)出願番号　特願平7－331666

(22)出願日　平成７年(1995)12月20日

(71)出願人　000005108
株式会社日立製作所
東京都千代田区神田駿河台四丁目６番地

(72)発明者　小山　昌宏
茨城県土浦市神立町502番地　株式会社日立製作所機械研究所内

(72)発明者　三宅　徳久
茨城県土浦市神立町502番地　株式会社日立製作所機械研究所内

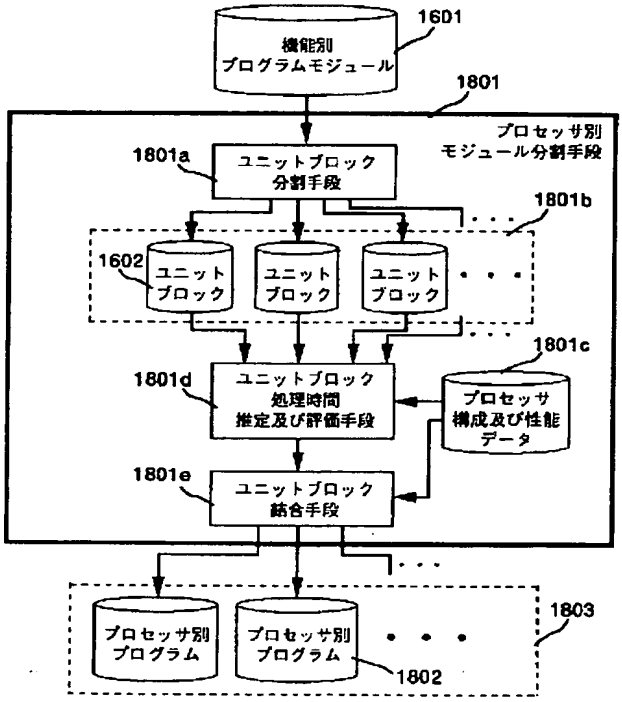
(74)代理人　弁理士　高崎　芳紘

(54)【発明の名称】　ＦＡシステムの制御装置と方法、制御プログラム生成方法

(57)【要約】

【課題】　ＦＡシステムの制御プログラムを一本化して記述し、自動的に各プロセッサに割り当て、そのプロセッサ用のプログラムに変換する。

【解決手段】　ＦＡシステムを構成する各セルの作業仕様を単一言語で記述し、シーケンス制御、自動機械の制御、プログラミング可能な周辺機器の制御を行う機能別プログラムモジュール群を生成し、これらの機能別プログラムモジュール１６０１の各々を複数のプロセッサ別プログラム１８０２からなるプロセッサ別プログラム群１８０３に分離する。この時、ユニットブロック処理時間推定及び評価手段１８０１ｄは、プロセッサ構成及び機能データ１８０１ｃにより各ユニットブロックの処理時間を評価し、複数のプロセッサにおける処理負荷の配分が最適になるように、ユニットブロック結合手段１８０１ｅがユニットブロック１６０２を結合する。



【特許請求の範囲】

【請求項１】　それぞれが、少なくとも自動機械と、前記自動機械に関連するプログラミング可能な周辺機器とを含む、複数のセルから構成されるＦＡシステムの制御装置であって、前記複数のセルに対して、それぞれ、
前記セル全体としての作業仕様を同一言語で記述したセル制御プログラムと、
前記セル制御プログラムから、前記自動機械と周辺機器の作業順序の制御に関するプログラムと、前記自動機械の動作制御に関するプログラムと、前記プログラミング可能な周辺機器の制御に関するプログラムとに分離変換する分離変換手段と、
前記分離変換手段により分離変換された各プログラムに基づいて、当該セルを構成する前記自動機械と前記プログラミング可能な周辺機器の動作をそれぞれ制御する複数のプロセッサ手段と、を備えており、前記分離変換手段は、さらに、少なくとも前記自動機械あるいは前記プログラミング可能な周辺機器の動作をそれぞれ制御する複数の前記プロセッサ手段における処理負荷の配分を行う処理負荷配分手段を備えていることを特徴とするＦＡシステムの制御装置。

【請求項２】　前記請求項１に記載のＦＡシステムの制御装置において、前記分離変換手段の処理負荷配分手段は、さらに、前記分離変換されたプログラムをユニット毎に分解する手段と、前記ユニット毎に分解されたプログラムの前記プロセッサ手段による処理時間を評価する手段とを備えていることを特徴とするＦＡシステムの制御装置。

【請求項３】　前記請求項２に記載のＦＡシステムの制御装置において、前記分離変換手段の処理負荷配分手段は、さらに、当該セルを構成する前記自動機械あるいは前記プログラミング可能な周辺機器の動作をそれぞれ制御する複数のプロセッサ手段の構成あるいは性能に関するデータを提供する手段を備えていることを特徴とするＦＡシステムの制御装置。

【請求項４】　前記請求項３に記載のＦＡシステムの制御装置において、前記ユニット毎に分解されたプログラムの処理時間を評価する手段は、前記データ提供手段からのプロセッサ手段の構成あるいは性能に関するデータを基にして評価するように構成されていることを特徴とするＦＡシステムの制御装置。

【請求項５】　前記請求項４に記載のＦＡシステムの制御装置において、前記分離変換手段の処理負荷配分手段は、さらに、前記分解手段によりユニット毎に分解されたプログラムを、前記プログラムの処理時間評価手段による評価により結合する手段を備えていることを特徴とするＦＡシステムの制御装置。

【請求項６】　それぞれが少なくとも自動機械と、前記自動機械に関連するプログラミング可能な周辺機器とを含む、複数のセルから構成されるＦＡシステムを制御するための制御方法において、
前記複数のセルの少なくとも１のセルを構成する機器をそれぞれ複数のプロセッサにより制御するため、前記セル全体としての作業仕様を同一言語で記述してセル制御プログラムを作成し、当該作成したセル制御プログラムから、前記自動機械と周辺機器の作業順序の制御に関するプログラムと、前記自動機械の動作制御に関するプログラムと、前記プログラミング可能な周辺機器の制御に関するプログラムとに分離変換し、前記分離変換した各プログラムに基づいて、当該セルを構成する前記自動機械と前記プログラミング可能な周辺機器の動作をそれぞれ前記複数のプロセッサにより制御するＦＡシステムの制御方法であって、当該プログラムの分離変換の際に、さらに、少なくとも前記自動機械あるいは前記プログラミング可能な周辺機器の動作を制御するそれぞれの前記複数のプロセッサにおける処理負荷の配分を行うことを特徴とするＦＡシステムの制御方法。

【請求項７】　前記請求項６に記載のＦＡシステムの制御方法において、前記プロセッサにおける処理負荷の配分は、前記それぞれの機器の動作を制御する複数のプロセッサにおいて、特定のプロセッサに処理負荷が集中しないように分配されることを特徴とするＦＡシステムの制御方法。

【請求項８】　前記請求項６に記載のＦＡシステムの制御方法において、前記複数のプロセッサにおける処理負荷の配分は、前記分離変換されたプログラムをユニット毎に分解し、前記それぞれの機器の制御を行う前記複数のプロセッサの構成あるいは性能に基づいて前記ユニット毎に分解されたプログラムの各々を評価し、当該評価の結果に基づいて、前記各プロセッサ毎に結合することにより行うことを特徴とするＦＡシステムの制御方法。

【請求項９】　それぞれが、少なくとも自動機械と、前記自動機械に関連するプログラミング可能な周辺機器とを含む、複数のセルから構成されるＦＡシステムにおいて、前記複数のセルの少なくとも１のセルを構成する機器をそれぞれ複数のプロセッサにより制御するための制御プログラムを生成するためのＦＡシステムの制御プログラム生成方法であって、
前記セル全体としての作業仕様を同一言語で記述してセル制御プログラムを作成し、
当該作成したセル制御プログラムから、前記自動機械と周辺機器の作業順序の制御に関するプログラムと、前記自動機械の動作制御に関するプログラムと、前記プログラミング可能な周辺機器の制御に関するプログラムとに自動的に分離変換し、そして、
少なくとも前記自動機械あるいは前記プログラミング可能な周辺機器の動作を制御するそれぞれの前記複数のプロセッサにおける処理負荷の配分を自動的に行う、ことを特徴とするＦＡシステムの制御プログラム生成方法。

【請求項１０】　前記請求項９に記載のＦＡシステムの

制御プログラム生成方法において、前記複数のプロセッサにおける処理負荷の配分は、前記分離変換されたプログラムをユニット毎に分解し、前記それぞれの機器の制御を行う前記複数のプロセッサの構成あるいは性能に基づいて前記ユニット毎に分解されたプログラムの各々を評価し、そして、当該評価の結果に基づいて、前記各プロセッサ毎に結合することにより行うことを特徴とするＦＡシステムの制御プログラム生成方法。

【請求項１１】　ロボットを含む自動機械と視覚センサなどのプログラミング可能な周辺機器とを含む複数機器の単位（セル）から構成されるＦＡシステムの制御方法において、

前記セル全体としての作業仕様を直接記述し、前記複数機器の作業順序の制御に関する情報と、前記複数機器の入出力の制御に関する情報と、前記自動機械の動作制御に関する情報と、前記プログラミング可能な周辺機器の制御に関する情報と、前記複数機器間の同期をとるための動作タイミングに関する情報とを有するセル制御プログラムから、セル制御プログラム変換手段によって、前記複数機器の作業順序に関する情報と、前記入出力の制御に関する情報を分離及び抽出して、前記複数機器の作業順序の制御と前記入出力の制御を行う処理が記述されたシーケンス制御プログラムを生成し、また、前記自動機械の動作制御と前記動作タイミングに関する情報を分離及び抽出して、前記自動機械の位置決め及び軌道の制御を行う処理が記述された自動機械制御プログラムを生成し、また、前記プログラミング可能な周辺機器の制御と前記動作タイミングに関する情報を分離及び抽出して、前記プログラミング可能な周辺機器の制御を行う処理が記述された周辺機器制御プログラムを生成し、このシーケンス制御プログラムと自動機械制御プログラムと周辺機器制御プログラムとを、それぞれシーケンス制御プログラム解釈実行手段と自動機械制御プログラム解釈実行手段と周辺機器制御プログラム解釈実行手段とに出力することを特徴とするＦＡシステムの制御方法。

【請求項１２】　ロボットなどの自動機械と視覚センサなどのプログラミング可能な周辺機器とを含む複数機器の単位（セル）から構成されるＦＡシステムの制御方法において、

前記セル全体としての作業仕様を直接記述し、前記複数機器の作業順序の制御に関する情報と、前記複数機器の入出力の制御に関する情報と、前記自動機械の動作制御に関する情報と、前記プログラミング可能な周辺機器の制御に関する情報と、前記複数機器間の同期をとるための動作タイミングに関する情報とを有するセル制御プログラムから、機能別モジュール分離変換手段によって、前記複数機器の作業順序に関する情報と、前記入出力の制御に関する情報を分離及び抽出して、前記複数機器の作業順序の制御と前記入出力の制御を行う処理が記述されたシーケンス制御プログラムモジュールを生成し、また、前記自動機械の動作制御と前記動作タイミングに関する情報を分離及び抽出して、前記自動機械の位置決め及び軌道の制御を行う処理が記述された自動機械制御プログラムモジュールを生成し、また、前記プログラミング可能な周辺機器の制御と前記動作タイミングに関する情報を分離及び抽出して、前記プログラミング可能な周辺機器の制御を行う処理が記述された周辺機器制御プログラムモジュールを生成し、さらにこのシーケンス制御プログラムモジュールと自動機械制御プログラムモジュールと周辺機器制御プログラムモジュールとを、それぞれシーケンス制御プロセッサ別モジュール分割手段と自動機械制御プロセッサ別モジュール分割手段と周辺機器制御プロセッサ別モジュール分割手段によって、前記セルを制御するプロセッサの構成と性能に基づいて、各プロセッサの処理負荷の配分が最適となるようなプログラムモジュールに分割し、さらにこれらの分割したプログラムモジュールを各プロセッサが実行可能な形式のプログラムに変換して、シーケンス制御プロセッサ用プログラム群と自動制御プロセッサ用プログラム群と周辺装置制御プロセッサ用プログラム群とを生成し、これらの各プロセッサ用プログラム群を、それぞれを解釈実行するプロセッサ群に対して出力することを特徴とするＦＡシステムの制御方法。

【請求項１３】　前記セル制御プログラムは、ペトリネットによって前記複数機器の動作状態の遷移を表現し、これに前記複数機器の動作内容を定義するコマンド列と、前記動作状態の終了を定義する条件式とを付加したプログラムであることを特徴とする請求項１２記載のＦＡシステムの制御方法。

【請求項１４】　前記セル制御プログラムは、前記ペトリネットを編集するウィンドウと、前記コマンド列を編集するウィンドウと、前記条件式を編集するウィンドウとを含むグラフィカル・ユーザ・インタフェースを備えたセル制御プログラム編集手段によって作成することを特徴とする請求項１３記載のＦＡシステムの制御方法。

【請求項１５】　前記セル制御プログラムは、前記ペトリネットの構造と一対一に対応するプログラムコードとして表現し、記憶することを特徴とする請求項１３記載のＦＡシステムの制御方法。

【請求項１６】　前記シーケンス制御プログラムモジュールと前記自動機械制御プログラムモジュールと前記周辺機器制御プログラムモジュールは、前記複数機器の動作タイミングの同期をとるための情報として、前記動作状態の整理番号を示す命令コードを含むことを特徴とする請求項１３記載のＦＡシステムの制御方法。

【請求項１７】　前記セル制御プログラムは、前記複数機器のうち、あるまとまった機器群で、その機器群どうしが互いに並行動作可能なもの（ユニット）の動作シーケンスをペトリネットの一つのモジュールとして記述し、これらのモジュールどうしを同期をとるためのプレ

ースによって結合することを特徴とする請求項１３記載のＦＡシステムの制御方法。

【請求項１８】　前記セル制御プログラムは、前記モジュールを一つのプログラムブロック（ユニットブロック）とした構造を有するプログラムコードとして表現することを特徴とする請求項１７記載のＦＡシステムの制御方法。

【請求項１９】　前記シーケンス制御プロセッサ別モジュール分割手段と前記自動機械制御プロセッサ別モジュール分割手段と前記周辺機器制御プロセッサ別モジュール分割手段は、前記ユニットブロックを単位としてプログラムモジュールの分割を行うことを特徴とする請求項１８記載のＦＡシステムの制御方法。

【請求項２０】　前記シーケンス制御プロセッサ別モジュール分割手段と前記自動機械制御プロセッサ別モジュール分割手段と前記周辺機器制御プロセッサ別モジュール分割手段は、前記ユニットブロックの対象とするプロセッサによる処理時間を推定し、その処理時間が制御上必要とされる基準を満たす範囲で前記ユニットブロックを結合し、前記シーケンス制御プロセッサ用プログラム群と前記自動制御プロセッサ用プログラム群と前記周辺装置制御プロセッサ用プログラム群とを生成することを特徴とする請求項１９記載のＦＡシステムの制御方法。

【請求項２１】　前記シーケンス制御プロセッサ用プログラム群と前記自動制御プロセッサ用プログラム群と前記周辺装置制御プロセッサ用プログラム群は、共有バスを介して結合されるシーケンス制御プロセッサ群と自動機械制御プロセッサ群と周辺機器制御プロセッサ群によって解釈実行することを特徴とする請求項１２記載のＦＡシステムの制御方法。

【請求項２２】　前記シーケンス制御プロセッサ群と前記自動機械制御プロセッサ群と前記周辺機器制御プロセッサ群とは、前記共有バスに接続された共有データ記憶手段に格納される前記複数機器の動作タイミングの同期をとるための情報にアクセス可能であることを特徴とする請求項２１記載のＦＡシステムの制御方法。

【請求項２３】　前記シーケンス制御プロセッサ用プログラム群と前記自動制御プロセッサ用プログラム群と前記周辺装置制御プロセッサ用プログラム群は、それぞれネットワークで接続された複数のプログラマブルコントローラと自動機械コントローラと周辺機器コントローラであることを特徴とする請求項１２記載のＦＡシステムの制御方法。

【請求項２４】　前記プログラマブルコントローラと前記自動機械コントローラと前記周辺機器コントローラは、前記ネットワークを介して、前記複数機器の動作タイミングの同期をとるための情報を互いに交換し、共有し合うことを特徴とする請求項２３記載のＦＡシステムの制御方法。

【請求項２５】　前記シーケンス制御プロセッサ用プログラム群と前記自動制御プロセッサ用プログラム群と前記周辺装置制御プロセッサ用プログラム群は、共有バスを介して結合されるシーケンス制御プロセッサ群と自動機械制御プロセッサ群と周辺機器制御プロセッサ群とから構成されるマルチプロセッサ型コントローラを、ネットワークによって複数個接続したものであることを特徴とする請求項１２記載のＦＡシステムの制御方法。

【請求項２６】　前記マルチプロセッサ型コントローラは、前記ネットワークを介して、前記複数機器の動作タイミングの同期をとるための情報を互いに交換し、共有し合うことを特徴とする請求項２５記載のＦＡシステムの制御方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、ロボットを含むＦＡシステムの制御装置と方法に係り、特に、ロボットなどの自動機械と視覚センサなどのプログラミング可能な周辺機器を含む複数機器によって構成されるＦＡシステムの制御装置と方法、さらには、かかるシステムにおける制御プログラムの生成方法に関する。

【０００２】

【従来の技術】ロボットなどの自動機械と、視覚センサなどのプログラミング可能な周辺機器とを含む複数の機器によって構成されるＦＡ（Factory Automation）システムは、既に広く知られ、かつ、実用化されている。そして、かかる従来のＦＡシステム及びその制御方法は、一般に、各種機器の入出力（以下、「Ｉ／Ｏ」と略記）を制御するプログラマブルコントローラと、ロボットを制御するロボットコントローラと、画像処理装置などを、パラレルＩ／Ｏ、または、ネットワークによって接続し、ＦＡシステムを構成する各機器の制御を行うというものであった。そして、かかる従来技術になるＦＡシステムの制御においては、各機器の制御対象毎に異なる制御装置（例えば、プログラマブルコントローラ、ロボットコントローラ、画像処理装置など）を組み合わせて使用し、しかも、制御装置毎に異なるプログラミング言語を使用していた。

【０００３】このため、ＦＡシステムを構築する場合、幾つもの異なったプログラム言語を習得する必要があり、一つのＦＡシステムを制御するためには、このような異なったプログラム言語を駆使して、互いに関連し合う複数のプログラムを作成しなければならなかった。また、ＦＡシステムの機能が高度化するにつれて、ＦＡシステムの制御装置の構成も複雑となり、そのプログラムの作成には多大な労力を要し、その際にも、上記異なるプログラミング言語の使用が、やはり、ＦＡシステムを制御するプログラムの開発効率を向上させる上で、大きな阻害要因となっていた。

【０００４】ところで、従来、上記のような課題を解決するものとして、例えば、特開平７－８４７６８号公報

によれば、ＦＡ用機器を作動させるプログラムを一つに統合した形で作成するＦＡシステムの制御方法が記載されている。また、やはり上記のような課題を解決するため、例えば、特開平７－７２９２０号公報によれば、ＦＡシステムの制御プログラムを一括して記述するＦＡシステムの制御方法が既に提案されている。

【０００５】すなわち、上記従来技術の前者である特開平７－８４７６８号公報に記載されているＦＡシステムの制御方法は、ロボットや画像処理装置などのＦＡ用機器を一つのプログラミング言語で統一的に作動させるための統合プログラムを作成して、これをプログラマブルコントローラ、ロボットコントローラ、画像処理装置で実行されるプログラムに分解するものである。なお、ここで述べられている統合プログラムは、従来のプログラマブルコントローラのラダープログラムに、ロボットプログラムと画像処理プログラムをサブルーチンのような形で付加した、いわゆる拡張型ラダープログラムなどであり、これをラダープログラム、ロボットプログラム、画像処理プログラムの三つに分解する方法をとっている。

【０００６】また、上記後者の従来技術である特開平７－７２９２０号公報に記載されているＦＡシステムの制御方法は、ＦＡシステムの作業を一括して記述した制御プログラムから、シーケンス制御プログラムとロボット制御プログラムを自動的に生成するものである。ここで述べられているＦＡシステムの制御プログラムは、ペトリネットを応用したプログラミング言語により、ＦＡシステム全体としての作業仕様を明示的に表現したものである。

【０００７】

【発明が解決しようとする課題】しかしながら、上記従来技術により提案されるＦＡシステムの制御方法や制御プログラムは、ＦＡシステムの機能の高度化に伴うプログラムの開発の効率を向上させる上ではその効果を発揮するものの、以下に指摘する問題点については言及されていなかった。

【０００８】すなわち、ＦＡシステムの機能が高度化するに伴い、複数の制御装置または制御用プロセッサによる分散協調化が進んだ場合、上記二つの従来技術により提案されるＦＡシステムの制御方法や制御プログラムのように、単に、プログラムをシーケンス制御やロボット制御などのような機能別に分離するだけでなく、さらに、制御装置や制御プロセッサの構成と性能に応じた最適な処理負荷の配分になるように、各制御装置毎またはプロセッサ毎のプログラムの分割を行う必要性が生じる。例えば、シーケンス制御プロセッサ、ロボット制御プロセッサ、画像処理プロセッサのそれぞれが複数個組み合わされたような制御システムにおいては、各プロセッサに対する最適なプログラム分割を自動的に行うことができれば、複雑なＦＡシステムを構築する際にも、ユーザは制御システムのハードウェア構成を意識することなく、より効率のよいＦＡシステムの制御プログラムを一本化して作成することができることとなる。

【０００９】そこで、本発明では、上記の従来技術における問題点に鑑み、さらには、上記発明者等による従来技術に対する認識に基づいて、すなわち、ＦＡシステムの制御プログラムを一本化して作成することができるだけではなく、さらに、作成される制御プログラムによりＦＡシステムをより効率的に制御することを可能にするＦＡシステムの制御装置と方法、さらには、かかるシステムにおける制御プログラムの生成方法を提供することを目的とする。

【００１０】

【課題を解決するための手段】上記の本発明の目的を達成するため、本発明によれば、それぞれが、少なくとも自動機械と、前記自動機械に関連するプログラミング可能な周辺機器とを含む、複数のセルから構成されるＦＡシステムの制御装置であって、前記複数のセルに対して、それぞれ、前記セル全体としての作業仕様を同一言語で記述したセル制御プログラムと、前記セル制御プログラムから、前記自動機械と周辺機器の作業順序の制御に関するプログラムと、前記自動機械の動作制御に関するプログラムと、前記プログラミング可能な周辺機器の制御に関するプログラムとに分離変換する分離変換手段と、前記分離変換手段により分離変換された各プログラムに基づいて、当該セルを構成する前記自動機械と前記プログラミング可能な周辺機器の動作をそれぞれ制御する複数のプロセッサ手段とを備えており、前記分離変換手段は、さらに、少なくとも前記自動機械あるいは前記プログラミング可能な周辺機器の動作をそれぞれ制御する複数の前記プロセッサ手段における処理負荷の配分を行う処理負荷配分手段を備えているＦＡシステムの制御装置が提起されている。

【００１１】また、本発明によれば、やはり上記の本発明の目的を達成するため、それぞれが、少なくとも自動機械と、前記自動機械に関連するプログラミング可能な周辺機器とを含む、複数のセルから構成されるＦＡシステムを制御するための制御方法において、前記複数のセルの少なくとも１のセルを構成する機器をそれぞれ複数のプロセッサにより制御するため、前記セル全体としての作業仕様を同一言語で記述してセル制御プログラムを作成し、当該作成したセル制御プログラムから、前記自動機械と周辺機器の作業順序の制御に関するプログラムと、前記自動機械の動作制御に関するプログラムと、前記プログラミング可能な周辺機器の制御に関するプログラムとに分離変換し、前記分離変換した各プログラムに基づいて、当該セルを構成する前記自動機械と前記プログラミング可能な周辺機器の動作をそれぞれ前記複数のプロセッサにより制御するＦＡシステムの制御方法であって、当該プログラムの分離変換の際に、さらに、少な

くとも前記自動機械あるいは前記プログラミング可能な周辺機器の動作を制御するそれぞれの前記複数のプロセッサにおける処理負荷の配分を行うＦＡシステムの制御方法が提案されている。

【００１２】さらに、本発明によれば、やはり上記の本発明の目的を達成するため、それぞれが、少なくとも自動機械と、前記自動機械に関連するプログラミング可能な周辺機器とを含む、複数のセルから構成されるＦＡシステムにおいて、前記複数のセルの少なくとも１のセルを構成する機器をそれぞれ複数のプロセッサにより制御するための制御プログラムを生成するためのＦＡシステムの制御プログラム生成方法であって、前記セル全体としての作業仕様を同一言語で記述してセル制御プログラムを作成し、当該作成したセル制御プログラムから、前記自動機械と周辺機器の作業順序の制御に関するプログラムと、前記自動機械の動作制御に関するプログラムと、前記プログラミング可能な周辺機器の制御に関するプログラムとに自動的に分離変換し、そして、少なくとも前記自動機械あるいは前記プログラミング可能な周辺機器の動作を制御するそれぞれの前記複数のプロセッサにおける処理負荷の配分を自動的に行うＦＡシステムの制御プログラム生成方法が提案されている。

【００１３】加えて、本発明によれば、上記の目的は、ロボットを含む自動機械と視覚センサなどのプログラミング可能な周辺機器とを含む複数機器の単位（セル）から構成されるＦＡシステムの制御方法において、前記セル全体としての作業仕様を直接記述し、前記複数機器の作業順序の制御に関する情報と、前記複数機器の入出力の制御に関する情報と、前記自動機械の動作制御に関する情報と、前記プログラミング可能な周辺機器の制御に関する情報と、前記複数機器間の同期をとるための動作タイミングに関する情報とを有するセル制御プログラムから、機能別モジュール分離変換手段によって、前記複数機器の作業順序に関する情報と、前記入出力の制御に関する情報を分離及び抽出して、前記複数機器の作業順序の制御と前記入出力の制御を行う処理が記述されたシーケンス制御プログラムモジュールを生成し、また、前記自動機械の動作制御と前記動作タイミングに関する情報を分離及び抽出して、前記自動機械の位置決め及び軌道の制御を行う処理が記述された自動機械制御プログラムモジュールを生成し、また、前記プログラミング可能な周辺機器の制御と前記動作タイミングに関する情報を分離及び抽出して、前記プログラミング可能な周辺機器の制御を行う処理が記述された周辺機器制御プログラムモジュールを生成し、さらにこのシーケンス制御プログラムモジュールと自動機械制御プログラムモジュールと周辺機器制御プログラムモジュールとを、それぞれシーケンス制御プロセッサ別モジュール分割手段と自動機械制御プロセッサ別モジュール分割手段と周辺機器制御プロセッサ別モジュール分割手段によって、前記セルを制御するプロセッサの構成と性能に基づいて、各プロセッサの処理負荷の配分が最適となるようなプログラムモジュールに分割し、さらにこれらの分割したプログラムモジュールを各プロセッサが実行可能な形式のプログラムに変換して、シーケンス制御プロセッサ用プログラム群と自動制御プロセッサ用プログラム群と周辺装置制御プロセッサ用プログラム群とを生成し、これらの各プロセッサ用プログラム群を、それぞれを解釈実行するプロセッサ群に対して出力するＦＡシステムの制御方法によって達成される。

【００１４】すなわち、本発明では、ＦＡシステムを構成する、一つのまとまった複数機器からなる単位要素（以下、セルとする）を構築する際に、セルに含まれるロボットなどの自動機械と視覚センサなどのプログラミング可能な周辺機器とその他の機器を制御するプログラムをセル制御プログラムとして一本化して記述し、さらに、このセル制御プログラムから、ＦＡシステムの制御装置のプロセッサの構成と性能に応じて、各プロセッサの処理負荷の配分が最適となるように、各プロセッサ用のプログラムを自動的に生成することで、従来のように制御装置の構成に応じて複数のプログラムを作成したり、幾つものプログラミング言語を習得したりする必要もなく、その結果として、ＦＡシステムの制御プログラムの開発効率を向上することができるＦＡシステムの制御方法を提供することが可能となる。

【００１５】

【発明の実施の形態】以下、添付の図面を参照して、本発明の実施の形態について詳細に説明する。図２は、本発明に係るＦＡシステムの制御装置における処理プロセスを模式的に示すブロック図である。一般に、ＦＡシステムは、自動機器であるロボットを含む複数の機器から構成され、これら複数の機器は、それぞれ、複数の機器の単位としての「セル」と呼ばれている。この図２の場合には、制御対象であるＦＡシステムの機器であるセル１１６は、例えば、２体のロボット１１６ａ、１１６ｂと、視覚センサ１１６ｃ、１１６ｄと、そして、その他の周辺機器１１６ｅ、１１６ｆなどから構成されている。

【００１６】かかるＦＡシステムのセル１１６の作業を制御するプログラミングを作成する場合、まず、このセル１１６全体としての作業仕様を記述したセル制御プログラム１０２を、コンピュータなどにより構成される、いわゆる、セル制御プログラム編集手段１０１によって作成する。なお、このセル制御プログラム編集手段１０１は、例えば、セル１１６の制御装置に接続された端末（図３に示す）の画面上において、セル１１６の各機器の動作シーケンスをペトリネットの形式で記述することができるグラフィカル・ユーザ・インタフェースを有するものである。また、セル制御プログラム１０２のペトリネット表現については、後に具体例を用いて説明す

る。

【００１７】セル制御プログラム１０２には、（１）ロボット１１６ａ、１１６ｂの動作制御に関する情報、（２）視覚センサ１１６ｃ、１１６ｄの画像処理に関する情報、（３）その他の周辺装置に接続されるＩ／Ｏの制御に関する情報、（４）これら各機器の動作順序と各機器間の動作タイミングの同期に関する情報が、セル１１６全体の作業仕様として、一種類のプログラミング言語（以下、セル制御言語とする）によって記述されている。このようにセル１１６の作業仕様を一つのプログラムとして記述することで、セル１１６全体としての動作シーケンスを明示的に表現することができる。なお、セル制御言語の仕様は、前記の従来技術、特開平７－７２９２０号公報に記載されているものに拡張を加えたものであり、シーケンス表現にペトリネットを応用しているところなど、基本的な部分は共通の手法を用いている。

【００１８】このセル制御プログラム１０２は、まず、機能別モジュール分離変換手段１０３によって、（ａ）シーケンス制御プログラムモジュール１０４と、（ｂ）ロボット制御プログラムモジュール１０５と、そして、（ｃ）画像処理プログラムモジュール１０６の３個のモジュールに分離及び変換される。ここで、上記のシーケンス制御プログラムモジュール１０４とは、セル１１６全体としてのシーケンス制御と、各種周辺機器１１６ｅ、１１６ｆなどのＩ／Ｏの制御に関するプログラムモジュールであり、ロボット制御プログラムモジュール１０５とは、ロボット１１６ａ、１１６ｂの動作制御に関するモジュールであり、そして、画像処理プログラムモジュール１０６とは、視覚センサ１１６ｃ、１１６ｄの画像処理に関するプログラムである。なお、以下の説明では、これらのシーケンス制御、ロボット制御、画像処理などのように制御機能別に分類されたプログラムモジュールのことを、場合により、まとめて機能別プログラムモジュールと呼ぶ。

【００１９】これら機能別プログラムモジュールは、それぞれが処理される制御用プロセッサ（プログマブルコントローラ、ロボットコントローラなどの独立したコントローラも、それぞれ広義の制御用プロセッサと考えるものとする）の構成に応じたプログラムモジュールに変換される。すなわち、本発明によれば、セル１１６の制御装置が複数の制御用プロセッサから構成されることを想定して、これらの各プロセッサの処理負荷の配分が最適になるように、機能別プログラムモジュールをプロセッサ別に分割することで、最終的に各プロセッサで実行されるプロセッサ別プログラムを生成する。具体的には、シーケンス制御プログラムモジュール１０４から、シーケンス制御プロセッサ別モジュール分割手段１０７によって、シーケンス制御プロセッサ別プログラム群１１０を生成する。また、ロボット制御プログラムモジュール１０５から、ロボット制御プロセッサ別モジュール分割手段１０８によって、ロボット制御プロセッサ別プログラム群１１１を生成する。また、画像処理プログラムモジュール１０６から、画像処理プロセッサ別モジュール分割手段１０９によって、画像処理プロセッサ別プログラム群１１２を生成する。

【００２０】そして、これらシーケンス制御プロセッサ別プログラム群１１０、ロボット制御プロセッサ別プログラム群１１１、画像処理プロセッサ別プログラム群１１２は、これら各プロセッサ用プログラム群１１０、１１１、１１２を解釈実行するプロセッサに対して出力される。すなわち、シーケンス制御プロセッサ用プログラム群１１０はシーケンス制御プロセッサ群１１３に対して、ロボット制御プロセッサ用プログラム１１４はロボット制御プロセッサ群１１４に対して、画像処理プロセッサ用プログラム群１１２は画像処理プロセッサ群１１５に対して出力され、それぞれのプロセッサがこれらのプログラムを解釈実行することで、セル１１６は制御される構成となっている。

【００２１】図３は、上記図２に示したＦＡシステムの制御装置の構成要件のうち、セル制御プログラム編集手段１０１における端末の画面表示の一例を示したものである。図示のように、このセル制御プログラム編集手段１０１は、セル制御プログラム１０２における動作シーケンスを、いわゆる、ペトリネットの形式で編集するためのグラフィカル・ユーザ・インタフェースを有している。このセル制御プログラム１０２の記述言語であるセル制御言語は、セル１１６の動作シーケンスの記述に対して、ペトリネットの図的表現形式を応用した一種のグラフィック言語である。この図３の例では、ペトリネットのプレース（図中、「○」で表記）をセル１１６を構成する機器の動作状態、トランジション（図中、「－」で表記）を次の動作状態に遷移するときの遷移条件、トークン（図中、表記せず）の置かれたプレースを現在の活性化された状態（以下、活性化状態とする）と定義し、各プレースには高々一個のトークンしか存在しないセーフペトリネットを考えている。

【００２２】なお、図３のネット編集ウィンドウ２０１では、マウス等のポインティングデバイスを用いて、画面上でグラフィックイメージのまま、ペトリネットを編集することができる。コマンド編集ウィンドウ２０２は、ネット編集ウィンドウ２０１の中でマウスのカーソルが指示している編集中の状態Ｓ１０６における機器の動作（ロボットの動作命令、Ｉ／ＯのＯＮ／ＯＦＦなど）やその他の処理を定義するためのコマンドを編集するためのウィンドウである。状態終了条件編集ウィンドウ２０３は、編集中のＳ１０６の正常終了の条件（ＯＫ条件）及び異常終了の条件（ＮＧ条件）を編集するためのウィンドウである。それぞれのウィンドウには、マウスやファンクションキーによるメニュー選択形式を利用した簡便な入力環境が用意されている。

【００２３】図４には、上記したセル制御プログラム編集手段１０１によって作成されたペトリネットの一例を示す。なお、この図４のペトリネットは、一例として、以下の図５及び図６に示す電子部品の挿入作業を表したものである。

【００２４】これら図５及び図６に示す作業は、プリント基板４０６にＩＣ等の電子部品４０５を挿入するもので、まず、コンベアなどで搬送されてきたプリント基板４０６を位置決めし、パーツフィーダなどによって供給される部品４０５を部品挿入ロボット４０１の先端に設けられたハンド４０２により把持し、所定の位置へ挿入する。さらに、挿入の終了した部品４０５のリードを、プリント基板４０６の下方からアクセスするクリンチロボット４０７の先端のクリンチハンド４０８によって曲げ、部品４０５がプリント基板４０６から脱落するのを防止する。また、それぞれのロボット４０１、４０７のプリント基板４０６に対する位置補正のために、視覚センサ４０４、４０９によって位置ずれ計測を行う。図５は、このような部品挿入直前における各機器の状態を示し、図６は部品挿入直後の状態を示している。

【００２５】上記図４においてペトリネットで表されたセル全体の動作シーケンスは、大きく分けると、部品挿入ロボット４０１（以下、「ＲＯＢ１」と略記）のモジュール３０４、クリンチロボット４０７（以下、「ＲＯＢ２」と略記）のモジュール３０５、部品挿入用の視覚センサ４０４（以下、「ＣＡＭ１」と略記）のモジュール３０６、クリンチ用の視覚センサ４０９（以下、「ＣＡＭ２」と略記）のモジュール３０７の４個のモジュールから構成される。各モジュールは、あるまとまった機器群で、その機器群同士で互いに並行動作可能なもの（以下、このような機器群のことを「ユニット」とする）毎にまとめられたシーケンスであり、これらのモジュールは同期プレース３０３（図中、二重丸で表記）で接続されることで、互いに関連付けられる。

【００２６】次に、上記図４のプレース３０１は、ユニットの動作状態を示しており、これを状態プレースと呼ぶ。例えば、状態プレース３０１に付記された「Ｓ１０６」は、動作状態の整理番号を示す。この状態プレース３０１における動作の具体的な内容は、ロボット命令やＩ／Ｏ制御命令などのコマンド列として定義する。また、トランジション３０２は、ある状態から次の状態に遷移するための条件を示している。トランジション３０２の遷移条件は、ペトリネットの接続関係と各状態の終了条件によって決まる。なお、このトランジション３０２に付記されている「ＯＫ１」は、このトランジション３０２への入力となる状態の正常終了を意味し、この正常終了の判定条件は、状態終了条件の式として定義される。同期プレース３０３は、モジュール間の動作タイミングの同期を定義するものであり、例えば、「ＲＥＱ６」は、状態Ｓ１０５が正常終了し、かつ、状態Ｓ２０４が正常終了したとき、次の状態Ｓ１０６への遷移条件が満たされる、という二つの状態終了の同期をとる。なお、図４の状態プレース３０１の横には、その状態における動作の内容を示すコメントを付記している。

【００２７】上記図４に示す動作シーケンスの概略を以下に説明する。まず、ＲＯＢ１が、ハンド４０２で把持した部品４０５を、位置決めされたプリント基板４０６の上の所定の位置（挿入点）まで移動する（Ｓ１００）と、ＣＡＭ１がＲＯＢ１の目標位置（プリント基板４０６との相対的な位置）からの位置ずれを計測し（Ｓ１５１）、その位置ずれ量に基づいてＲＯＢ１は位置補正の動作を行う（Ｓ１０２）。さらに、ＲＯＢ１は、そのリストを所定の高さまで下降させた後、ハンド４０２を開いて部品１０３を解放する（Ｓ１０３）。この間にＲＯＢ２は、プリント基板４０６の直下の所定の位置（クリンチ点）まで移動（Ｓ２００）し、ＣＡＭ２によって位置ずれ計測（Ｓ２５１）した結果に基づいてＲＯＢ２は位置補正を行い（Ｓ２０２）、さらに、そのリストを所定の高さまで上昇させる（Ｓ２０３）。ＲＯＢ１のハンド４０２に設けられたセンサ（図示せず）によって部品４０５の解放が検知されると、ＲＯＢ１の手先に設けられたセンタプッシャ４０３が下降することで部品４０５を押し下げ、プリント基板４０６の所定の穴に部品４０５を深く挿入する（Ｓ１０４）。ＲＯＢ１の手先に設けられたセンサ（図示せず）によりセンタプッシャ４０３が所定の高さまで下降したことが検知されると、ＲＯＢ２のクリンチハンド４０８が閉じて、部品４０５のリードが折り曲げられる（Ｓ２０４）。クリンチハンド４０８に設けられたセンサ（図示せず）により、クリンチが完了したことが検知されると、クリンチハンド４０８が開かれ（Ｓ２０５）、ＲＯＢ１のセンタプッシャ４０３が上昇する（Ｓ１０６）。その後、ＲＯＢ１は挿入準備点まで移動し（Ｓ１０７）、ＲＯＢ２はリストを下降し（Ｓ２０６）、クリンチ準備点まで移動する（Ｓ２０７）。

【００２８】次に、図７と図８には、上記図３にも説明したセル制御プログラム編集手段１０１によって、図４のペトリネットを編集し、それをコード化したもの、すなわち、図２のセル制御プログラム１０２の一例を示す。図からも明らかなように、セル制御プログラム１０２は、２種類のブロックから構成されており、一つは図７に示すｃｅｌｌブロック（”ｃｅｌｌ”文に続くブロック）であり、もう一つは、図８に示すｄｅｆブロック（”ｄｅｆ”文に続くブロック）である。

【００２９】このセル制御プログラム１０２について、さらに、その詳細な内容を説明すると、図７のｃｅｌｌブロックには、ペトリネットの接続関係と各状態における機器の動作が定義されている。ｃｅｌｌラベル６０１には、状態の整理番号とその状態における動作主体であるユニットの種別とその整理番号が定義される。例え

ば、状態Ｓ１０６のユニットはＲＯＢ１であり、これは部品挿入ロボット４０１とそれに付随するハンド４０２等の周辺機器から構成されるユニットを示す。遷移条件式６０２は、ペトリネットの接続関係を定義する式であり、その状態の入力トランジションに入力として接続される状態の整理番号とその状態の終了条件を定義する。状態Ｓ１０６の入力トランジションは、状態Ｓ１０５の正常終了ＯＫ１と同期プレースＲＥＱ６のＯＫとに接続されている。コマンド列６０３は、その状態におけるユニットを構成する機器の動作とその他の処理を定義するコマンド列である。状態Ｓ１０６における処理は、ＲＥＳＥＴ　Ｙ１０１（出力Ｙ１０１のＯＦＦ、すなわちセンタプツシャ４０３の上昇）と、ＳＥＴ　ＴＤ１２１０．（オンディレイタイマＴＤ１２のセット）である。

【００３０】また、図８のｄｅｆブロックには、各状態の終了条件が定義される。ｄｅｆラベル７０１には状態の整理番号が定義され、定義式７０２には状態の正常終了（ＯＫ）と異常終了（ＮＧ）の条件式が定義される。状態Ｓ１０６の正常終了条件（ＯＫ１）は、Ｘ１０１＝ＯＦＦ（入力Ｘ１０１がＯＦＦ、すなわち部品４０５の接触検出のセンサがＯＦＦ）かつＴＤ１２＝ＯＦＦ（オンディレイタイマＴＤ１２がＯＦＦ、すなわちタイムアップしていない）である。状態Ｓ１０６の異常終了条件（ＮＧ１）は、Ｘ１０１＝ＯＮかつＴＤ１０２＝ＯＮ（オンディレイタイマＴＤ１０２がタイムアップしても、センサがＯＮのまま）である。なお、図中の「＝＝」は、条件式であることを示す。

【００３１】このように、上記図４にも示すように、セル制御プログラム１０２の記述にペトリネットの表現を応用することで、各機器の動作シーケンスが構造的かつ明示的に記述することができる。さらに、図７、図８のように、セル内の機器の制御プログラムをセル制御プログラム１０２として一本化し、しかも、各機器の動作シーケンスの明示性を持たせることで、ＦＡシステムのソフトウェアの開発者にとって、セル内の機器の制御装置の構成やプログラム言語の違いを意識せずに、セル全体としての作業仕様を直接記述する形で、セル制御プログラム１０２を容易に作成することが可能になる。すなわち、セル制御プログラム１０２の開発者は、従来のようにいくつかの異なったプログラム言語を習得する必要がなくなり、その結果、プログラムの開発効率が向上し、ソフトウェアとしての保守性も向上する。

【００３２】そして、本発明によれば、上記図７、図８のようなセル制御プログラム１０２から、機能別モジュール分離変換手段１０３によって、セルを構成する機器の動作順序の制御とこれらの機器に接続されたＩ／Ｏの制御を行うする処理が記述されたシーケンス制御プログラムモジュール１０４と、ロボット１１６ａ、１１６ｂの位置決め及び軌道の制御を行う処理が記述されたロボット制御プログラムモジュール１０５と、視覚センサ１１６ｃ、１１６ｄの画像に関する処理を記述した画像処理プログラムモジュール１０６を生成する。

【００３３】図９には、上記機能別モジュール分離変換手段１０３の処理プロセス示す。なお、セル制御プログラム１０２のｃｅｌｌブロックには、上記の図７及び図８に示したように、セル制御プログラム１０２のｃｅｌｌブロックには、ペトリネットにおける各プレースの接続関係、すなわちネット構造と、各状態における機器の動作が定義され、また、ｄｅｆブロックには、各状態の終了条件が定義される。

【００３４】図の機能別モジュール分離変換手段１０３の機能別データ抽出手段１０３ａは、まず、ｃｅｌｌブロックの内容からネット構造１０３ｂを、ｄｅｆブロックの内容から状態終了条件１０３ｃを抽出する。さらに、ｃｅｌｌブロックからＩ／Ｏ制御命令列１０３ｄを抜き出す。そして、シーケンス制御プログラム生成手段１０３ｉは、上記抽出したネット構造１０３ｂ、状態終了条件１０３ｃ、そして、Ｉ／Ｏ制御命令列１０３ｄから、シーケンス制御プログラムモジュール１０４を生成する。

【００３５】また、機能別データ抽出手段１０３ａは、ｃｅｌｌブロックの内容から、ロボット制御命令列１０３ｅとロボット動作タイミング１０３ｆに関する情報を抽出する。そして、ロボット制御プログラム生成手段１０３ｊは、これら抽出した情報に基づいて、ロボット制御プログラムモジュール１０５を生成する。

【００３６】同様に、機能別データ抽出手段１０３ａは、ｃｅｌｌブロックの内容から画像処理命令列１０３ｇと画像処理タイミング１０３ｈに関する情報を抽出し、さらに、画像処理プログラム生成手段１０３ｋは、これらの情報に基づいて画像処理プログラムモジュール１０６を生成する。

【００３７】図１０、図１１、及び、図１２には、上記図７、図８に例示したセル制御プログラム１０２から、上記図９に示した機能別モジュール分離変換手段１０３によって生成された、シーケンス制御プログラムモジュール１０４の一例を示す。なお、本例におけるシーケンス制御用のプログラミング言語は、ＩＦ～ＴＨＥＮ～形式のルール型の記述を基本とする。すなわち、各ルールには、各状態における機器の動作の実行ステータス、センサなどの外部からの入力信号、内部変数の値などについての評価式を条件とし、ここで所定の条件が満たされれば、その状態における機器の動作を終了し、次の段階の状態における機器の動作を起動する、という処理が記述される。例えば、図１１中に示したルール１００１は、状態Ｓ２０３が正常終了し、かつ同期プレースＲＥＱ５が正常終了すれば、状態Ｓ２０４の実行ステータスを実行中にし（ＥＸＥＣ　Ｓ２０４）、出力Ｙ２００とオンディレイタイマＴＤ２０をセットすることを表す。

【００３８】なお、シーケンス制御用のプログラミング

言語としては、本例のようなルール型のプログラミング言語の他にも、例えばラダー図などのプログラミング言語を用いることも可能であり、その場合、機能別モジュール分離変換手段１０３は、シーケンス制御プログラムモジュール１０４としてラダープログラムを生成することとなることは当業者にとっては当然であろう。

【００３９】図１３には、上記図７、図８のセル制御プログラム１０２から、上記機能別モジュール分離変換手段１０３によって生成された、ロボット制御プログラムモジュール１０５の一例を示す。この図１３の中に示したブロック１２０３は、状態Ｓ２０３（ＳＴＴ２０３）において、ＲＯＢ２のリストを１０ｍｍだけ上昇させる（ＭＯＶＥ　Ｌ，＊＋（０．，０．，１０．））ことを表す。ここで、ＳＴＴｎ（ｎは状態Ｓｎのｎと同じ）は、先に示したシーケンス制御プログラムモジュール１０４と共通する状態の整理番号（以下、状態ＮＯ．とする）を示す状態ラベルであり、この状態ＮＯ．を各プログラムの間で共有することによって、セル全体の状態遷移を制御することができる。

【００４０】図１４には、上記図７、図８のセル制御プログラム１０２から、上記機能別モジュール分離変換手段１０３によって生成された画像処理プログラムモジュール１０６の一例を示す。ここで生成される画像処理プログラムモジュール１０６は、ロボット制御プログラムモジュール１０５と同様の構成を持っており、ＳＴＴｎ文に続けて各種の画像処理命令列が記述されている。

【００４１】さらに、図１５～図２４には、上記機能別モジュール分離変換手段１０３の処理の詳細を説明するフローチャートを示す。まず、図１５は、上記機能別モジュール分離変換手段１０３における機能別モジュール分離変換処理の概略を示す。この機能別モジュール分離変換処理では、処理がスタートすると、まず、セル制御プログラムを解析して機能別データの抽出を行い（処理１１００ａ）、ここで得られたデータに基づいて、機能別モジュールを生成する。すなわち、シーケンス制御プログラム生成手段１０３ｉによってシーケンス制御プログラムモジュール１０４を生成し（処理１１００ｂ）、ロボット制御プログラム生成手段１０３ｊによってロボット制御プログラムモジュール１０５を生成し（処理１１００ｃ）、画像処理プログラム生成手段１０３ｋによって画像処理プログラムモジュール１０６を生成する（処理１１００ｄ）。

【００４２】図１６は、上記図１５に示した機能別モジュール分離変換処理の機能別データ抽出処理１１００ａの詳細を示す。この機能別データ抽出処理１１００ａでは、処理がスタートすると、セル制御プログラムのｃｅｌｌブロック、ｄｅｆブロックの順にコード解析を行い、それぞれ必要なデータ抽出を行う。まず、ｃｅｌｌブロックの解析では、各ロボット毎の動作タイミング１０３ｆとロボット制御命令列１０３ｅを格納するためのデータ領域の確保（処理１１０１ａ）、各視覚センサ毎の画像処理タイミング１０３ｈと画像処理命令列１０３ｇを格納するためのデータ領域の確保（処理１１０１ｂ）、Ｉ／Ｏ制御命令列１０３ｄを格納するためのデータ領域の確保（処理１１０１ｃ）を行う。次に、セル制御プログラムを、ペトリネットの１プレースに相当するブロック単位で、先頭から順に処理していき、プレースの接続関係、すなわち、ネット構造１０３ｂをコード解析によって抽出し（処理１１０１ｄ）、さらに、各プレースにおいて定義されるコマンド列を解析する（処理１１０１ｅ）。これらの解析は、次のプレースのブロックがなくなるまで繰り返す（処理１１０１ｆ）。ｄｅｆブロックの解析では、１プレース毎に状態の終了条件１０３ｃをコード解析によって抽出し（処理１１０１ｇ）、次のプレースのブロックがなくなるまで、これを繰り返して（処理１１０１ｈ）、終了する。

【００４３】次に、図１７及び図１８は、１プレース毎のネット構造解析処理１０１ｄの詳細を示す。この１プレース毎のネット構造解析処理１０１ｄでは、まず、１プレース毎のネット構造データ領域を確保し（処理１１０２ａ）、次に、ｃｅｌｌラベル６０１と遷移条件式６０２の解析を行う。図１７のｃｅｌｌラベルの解析では、まず、そのｃｅｌｌラベルが状態プレースのラベルか、あるいは、同期プレースのラベルかを調べ（処理１１０２ｂ）、その結果、状態プレースであれば、その状態ＮＯ．を抽出し、ネット構造データの一つとして格納する（処理１１０２ｃ）。さらに、ｃｅｌｌラベルからユニット種別とユニットＮＯ．を抽出し、ネット構造データとして格納する（処理１１０２ｄ、１１０２ｅ）。ここで、ユニット種別がロボットであれば（処理１１０２ｆ）、ロボット動作タイミング１０３ｆのデータとして先に抽出した状態ＮＯ．を格納する（処理１１０２ｇ）。また、ユニット種別が視覚センサであれば（処理１１０２ｈ）、画像処理タイミング１０３ｈのデータとして先の状態ＮＯ．を格納する（処理１１０２ｉ）。一方、ｃｅｌｌラベルが同期プレースのラベルであれば（処理１１０２ｂ）、その同期プレースの整理番号（以下、同期ＮＯ．）を抽出し、ネット構造データとして格納する（処理１１０２ｊ）。

【００４４】一方、図１８の遷移条件式の解析では、まず、遷移条件式６０２（上記図７の例では、Ｓ１０５：ＯＫ１　ＲＥＱ６：ＯＫ）から状態ＮＯ．（Ｓ１０５）と終了条件（ＯＫ１）を抽出し、これらをネット構造データとして格納する（処理１１０３ａ、１１０３ｂ）。ここで、同期条件があれば（処理１１０３ｃ）、同期ＮＯ．（ＲＥＱ６）を抽出し、ネット構造データとして格納する（処理１１０３ｄ）。さらに、別の遷移条件があれば、その遷移条件について同様の解析処理を行い、遷移条件がなければ解析処理を完了する（処理１１０３ｅ）。

【００４５】以上のように、上記図１７と図１８に示したネット構造解析処理により、与えられたセル制御プログラムと等価なペトリネットの各プレースの接続関係、すなわち、ネット構造１０３ｂの情報が得られる。

【００４６】次に、図１９は、上記１プレース毎のコマンド列解析１０１ｅの詳細を示す。ここでは、ｃｅｌｌラベル６０１（図７を参照）と遷移条件式６０２に続くコマンド列６０３の解析を行う。まず、コマンド列６０３のブロックから１コマンドを抽出し（処理１１０４ａ）、これがＩ／Ｏ制御命令であれば（処理１１０４ｂ）、Ｉ／Ｏ制御命令列データとして格納する（処理１１０４ｃ）。また、抽出したコマンドがロボット制御命令であれば（処理１１０４ｄ）、これをロボット制御命令列データとして格納する（処理１１０４ｅ）。抽出したコマンドが画像処理命令であれば（処理１１０４ｆ）、画像処理命令列データとして格納する（処理１１０４ｇ）。なお、次のコマンドがあれば、処理１１０４ａ以降を繰り返し、コマンドがなければ、解析処理を完了する（処理１１０４ｈ）。

【００４７】図２０は、上記１プレース毎の終了条件解析１０１ｇの詳細を示す。ここでは、セル制御プログラムのｄｅｆブロックに定義された各プレース（状態）の終了条件の解析を行う。まず、１プレース毎の終了条件データを格納する領域を確保する（処理１１０５ａ）。さらに、ｄｅｆラベル７０１から状態ＮＯ．（図８の例では、Ｓ１０６）を抽出する（処理１１０５ｂ）。次に、ｄｅｆラベル７０１に続くＯＫ条件の定義式（図８の例では、Ｘ１０１＝ＯＦＦ　＆　ＴＤ１２＝ＯＦＦ）を抽出し、これを終了条件データとして格納する（処理１１０５ｃ）。次のＯＫ条件がなくなるまで、この処理を繰り返す（処理１１０５ｄ）。さらに、ＮＧ条件の定義式（図８の例では、Ｘ１０１＝ＯＮ　＆　ＴＤ１２＝ＯＮ）を抽出し、これを終了条件データとして格納する（処理１１０５ｅ）。次のＮＧ条件がなくなるまで、この処理を繰り返す（処理１１０５ｆ）。以上の処理により、各状態の終了条件を得ることができる。

【００４８】図２１及び図２２は、上記シーケンス制御プログラム生成処理１００ｂの詳細を示す。ここでは、先の機能別データ抽出処理１００ａによって得られたネット構造１０３ｂ、状態終了条件１０３ｃ、Ｉ／Ｏ制御命令列１０３ｄのデータに基づいて、各ユニット毎にシーケンス制御プログラムを生成する。まず、各ユニット毎のプログラムバッファ領域を確保する（処理１１０６ａ）。次に、先に生成したネット構造データを１プレースずつ順に調べ、プレース毎にシーケンス制御プログラムを生成していく。

【００４９】そこで、まず、１プレース毎のネット構造データからプレース種別を獲得し（処理１１０６ｂ）、そのプレースが状態プレースであれば（処理１１０６ｃ）、その状態ＮＯ．（整理番号をｉとおく。以下、「＝ｉ」と略記）とユニットＮＯ．を獲得する（処理１１０６ｄ）。そのプレースが状態プレースでなければ、ネット構造データから同期ＮＯ．（＝ｉ）を獲得する（処理１１０６ｅ）。さらに、ネット構造データと終了条件データから、現在着目しているプレースに接続する、一つ前の状態プレースの状態ＮＯ．（＝ｊ）と終了条件の定義式を獲得する（処理１１０６ｆ）。この一つ前の状態の終了条件が、”ＣｏｍｐｅｌｅｔｅＮｏｒｍａｌｌｙ”であれば（処理１１０６ｇ）、”ＩＦ　Ｓｊ＝ＣｏｍｐｌｅｔｅＮｏｒｍａｌｌｙ”をプログラムバッファに出力し（処理１１０６ｈ）、”ＡｎｙＥｒｒｏｒＯｃｃｕｒ”であれば（処理１１０６ｉ）、”ＩＦ　Ｓｊ＝ＡｎｙＥｒｒｏｒＯｃｃｕｒ”をバッファに出力する（処理１１０６ｊ）。それ以外の定義式の場合、”ＩＦ　Ｓｊ＝Ｅｘｅｃｕｔｉｎｇ　＆　（定義式）”をバッファに出力する（処理１１０６ｋ）。

【００５０】ここで、現プレースが状態プレースであれば（処理１１０６ｌ）、現プレースに接続する同期プレースがあるかどうか調べ、同期プレースがあれば（処理１１０７ａ）、その同期ＮＯ．（＝ｋ）をネット構造データから獲得し（処理１１０７ｂ）、”＆　ＲＥＱｋ＝ＣｏｍｐｌｅｔｅＮｏｒｍａｌｌｙ”をバッファに出力する（処理１１０７ｃ）。以上の処理で、１プレースあたりのシーケンス制御プログラム、すなわち、ルール１００１（図１１を参照）の条件部が生成される。

【００５１】次に、ルール１００１の実行部として、まず、”－＞　ＴＥＲＭ　Ｓｊ”をバッファに出力する（処理１１０７ｄ）。ここで、現プレースに接続する同期プレースがあれば（処理１１０７ｅ）、”ＴＥＲＭ　ＲＥＱｋ”をバッファに出力する（処理１１０７ｆ）。さらに、”ＥＸＥＣ　Ｓｉ”をバッファに出力し（処理１１０７ｇ）、現プレース（Ｓｉ）のＩ／Ｏ制御命令列データの内容をバッファに出力する（処理１１０７ｈ）。また、現プレースが同期プレースであれば（処理１１０６ｌ）、”－＞　ＴＥＲＭ　Ｓｊ”と”ＥＸＥＣ　ＲＥＱｉ”をバッファに出力する（処理１１０７ｉ、１１０７ｊ）。以上の処理で、ルール１００１の実行部が生成される。この後、さらに、ネット構造データを調べ、次のプレースが見つかれば（処理１１０７ｋ）、処理１１０６ｂ以降を繰り返す。次のプレースがなければ、各ユニット毎のプログラムバッファを結合し、シーケンス制御プログラムモジュール１０４（図９を参照）を生成する（処理１１０７ｌ）。

【００５２】図２３は、上記ロボット制御プログラム生成処理１００ｃの詳細を示す。まず、各ユニット毎のプログラムバッファ領域を確保する（処理１１０８ａ）。次に、先の機能別データ抽出処理１００ａで得られたユニット毎のロボット動作タイミング１０３ｆのデータから、状態ＮＯ．（＝ｉ）を一つずつ獲得し（処理１１０８ｂ）、状態ラベル”ＳＴＴｉ”をプログラムバッファ

に出力する（処理１１０８ｃ）。さらに、上記の機能別データ抽出処理１００ａで得られたロボット制御命令列１０３ｅのデータがあれば（処理１１０８ｄ）、そのロボット制御命令列データの内容をバッファに出力する（処理１１０８ｅ）。ロボット制御命令列データがなければ、”ＮＯＰ”をバッファに出力する（処理１１０８ｆ）。以上の処理で１状態分のロボット制御プログラムが生成される。ここで、ロボット動作タイミングデータを調べ、次に続く状態が見つかれば（処理１１０８ｇ）、処理１１０８ｂ以降を繰り返す。次の状態がなければ、これで１ユニット分のロボット制御プログラムが生成されたことになる。さらに、次のユニットのロボット動作タイミングデータがあるかどうか調べ、次のユニットのデータがあれば（処理１１０８ｈ）、処理１１０８ｂ以降を繰り返す。次のユニットのデータがなければ、各ユニット毎のプログラムバッファを結合し、ロボット制御プログラムモジュール１０５（図９を参照）を生成する（処理１１０８ｉ）。

【００５３】図２４は、画像処理プログラム生成処理１００ｄの詳細を示す。なお、図からも明らかなように、ここでは、上記図２３と同様の処理により、図１５の機能別データ抽出処理１１００ａで得られた画像処理タイミング１０３ｈ（図９を参照）のデータと、画像処理命令列１０３ｇのデータから、画像処理プログラムモジュール１０６を生成する。

【００５４】次に、図２５は、セル制御プログラム１０２（図９を参照）からシーケンス制御プログラム１０４と、ロボット制御プログラムモジュール１０５とを生成する手順を、上記図７、図８に示したセル制御プログラムの一部を例にして、具体的に示すものである。

【００５５】この図２５の符号１４０１、１４０２は、それぞれ、上記セル制御プログラム１０２のｃｅｌｌブロック、ｄｅｆブロックの一部を示す。このｃｅｌｌブロック１４０１において、先に述べたｃｅｌｌラベル６０１（図７を参照）と状態遷移式６０２を合わせたものが、この図中のペトリネットの状態接続関係１４０１ａ、１４０１ｃである。この状態接続関係１４０１ａは、状態Ｓ１０６への入力トランジションが一つであり、そこに状態Ｓ１０５の正常終了ＯＫ１と同期プレースＲＥＱ６のＯＫが入力として接続されていることを示す。同様に、状態接続関係１４０１ｃは、状態Ｓ１０７のトランジションは一つで、そこに状態Ｓ１０６のＯＫ１が接続されていることを示す。

【００５６】また、ｄｅｆブロック１４０２には、各状態の終了条件が定義されており、状態Ｓ１０５の正常終了ＯＫ１の状態終了条件１４０２ａは、状態Ｓ１０５における動作がエラーなしで完了することを示す。同様に、状態Ｓ１０６のＯＫ１の状態終了条件１４０２ｃは、入力Ｘ１０１がＯＦＦであり、かつ、オンディレイタイマＴＤ１２がタイムアップしていないことを示す。

【００５７】これら状態接続関係１４０１ａと状態終了条件１４０２ａから、シーケンス制御プログラム１４０３におけるルールの条件部１４０３ａと、実行部の状態遷移命令１４０３ｂ（ＥＸＥＣ　Ｓ１０６など）が生成される（処理１４０５、１４０６）。すなわち、ルール条件部１４０３ａとして「もし、状態Ｓ１０５が正常終了であり、かつ同期プレースＲＥＱ６が満たされれば」が生成され、状態遷移命令１４０３ｂとして「状態Ｓ１０６を実行する」などが生成される。同様にして、状態接続関係１４０１ｃと状態終了条件１４０２ｃから、ルールの条件部１４０３ｄと、実行部の状態遷移命令１４０３ｅが生成される（処理１４０７、１４０８）。

【００５８】また、このとき同時に、状態接続関係１４０１ａ、１４０１ｃから、状態ＮＯ．すなわちロボット動作タイミング１０３ｆが抜き出され、ロボット制御プログラム１４０４の中の状態ラベル１４０４ａ、１４０４ｃが生成される（処理１４０９、１４１０）。

【００５９】さらに、ｃｅｌｌブロック１４０１のコマンド列から、Ｉ／Ｏ制御命令列１４０１ｂが抜き出されて、シーケンス制御プログラム１４０３のルールの後件部のコマンド列１４０３ｃが生成される（処理１４１１）。さらに、ｃｅｌｌブロック１４０１のコマンド列のうち、ロボット制御命令列１４０１ｄが抜き出され、ロボット制御プログラム１４０４のコマンド列１４０４ｂ、１４０４ｄが生成される（処理１４１２、１４１３）。

【００６０】図２６は、セル制御プログラム１０２（図９を参照）から、シーケンス制御プログラム１０４と画像処理プログラムモジュール１０６を生成する手順を、やはり、図７、図８に示したセル制御プログラムの一部を例にして、具体的に示すものである。すなわち、上記の図２５と同様に、セル制御プログラム１０２のｃｅｌｌブロック１５０１と、ｄｅｆブロック１５０２とから、シーケンス制御プログラム１５０３と、画像処理プログラム１５０４とを生成する手順を示している。

【００６１】以上のようにして、機能別モジュール分離変換手段１０３によって生成される機能別プログラムモジュールは、共通のプログラム構造を持ち、その構造を図２７に示す。すなわち、図２７に示すように、そのプログラム構造において、機能別プログラムモジュール１６０１は、いくつかのユニットブロック１６０２によって構成され、さらに、このユニットブロック１６０２は、状態ブロック１６０３によって構成される。このユニットブロック１６０２は、ユニットラベル（＃ＵＮＩＴｎ、＃ＲＯＢｎ、＃ＣＡＭｎなど、ｎはユニットの整理番号）に続くプログラムのブロックである。また、状態ブロック１６０３は、シーケンス制御プログラムではルール１００１に相当し、ロボット制御プログラムと画像処理プログラムでは状態ラベル（ＳＴＴｎ、ｎは状態ＮＯ．）に続くブロックに相当する。

【００６２】例えば、図１１に例示したシーケンス制御プログラムは、ＲＯＢ１のユニットブロック１６０２であり、ルール１００１は状態Ｓ２０３の状態ブロック１６０３に相当する。また、図１４のロボット制御プログラムは、ＲＯＢ１のユニットブロック１２０１と、ＲＯＢ２のユニットブロック１２０２から構成されており、また、図中のブロック１２０３は、図２７の状態Ｓ２０３の状態ブロック１６０３に相当する。また、図１４の画像処理プログラムも、図１３と同様、図２７に示す、ＣＡＭ１のユニットブロック１６０２と、ＣＡＭ２のユニットブロック１６０２とから構成されている。

【００６３】この図２７に示した機能別プログラムモジュール１６０１のブロック構造は、この機能別プログラムモジュール１６０１を分割することで得られるプロセッサ別プログラムのコード実行手順と密接な関係がある。すなわち、プログラム実行時には、あるユニットブロック１６０２の中のうち、現在活性化されている状態ブロック１６０３が選び出され、そのブロックに記述された命令列が順番に実行される。さらに、別のユニットブロック１６０２においても、同様の処理が行われる。ここで、処理中のユニットブロック１６０２を指すユニットラベルの番号（＃ＵＮＩＴｎのｎなど）は、ユニットそのものの整理番号（以下、ユニットＮＯ．とする）と同じものであり、このユニットＮＯ．は、プログラムの実行時にプログラム制御を行うための情報（以下、プログラム制御用データとする）として管理される。また、機器の現在の状態、すなわち活性化状態における動作が記述された状態ブロック１６０３を指す状態ラベルの番号は、状態ＮＯ．として管理される。

【００６４】次に、図２８には、このプロセッサ別プログラムの実行手順を示す。同図に示す処理フローにおいて、まず、現在のユニットＮＯ．をプログラム制御用データから読み込み（処理１７０１）、そのユニットＮＯ．と同じ番号を持つユニットラベルへジャンプする（処理１７０２）。次に、活性化状態の状態ＮＯ．を読み込み（処理１７０３）、その状態ラベルへジャンプする（処理１７０４）。この状態ラベルに続く命令列を順番に実行し（処理１７０５、１７０６）、その実行が終了すると次のユニットブロック１６０２を処理するためにユニットＮＯ．を更新し（処理１７０７）、再びフローの最初に戻る。このように、プログラムはユニットブロック１６０２を単位として実行されるので、機能別プログラムモジュール１６０１はこのユニットブロック１６０２を単位として、さらに分割することができる。また、先に述べたプロセッサ別プログラムは、機能別プログラムモジュール１６０１を、上記ユニットブロック１６０２を単位として分解することによって、生成される。

【００６５】続いて、図１には、上記の機能別プログラムモジュール１６０１から、プロセッサ別プログラム群１８０３を生成する、いわゆる、プロセッサ別モジュール分割手段１８０１の処理プロセスを示す。このプロセッサ別モジュール分割手段１８０１では、まず、機能別プログラムモジュール１６０１をユニットブロック分割手段１８０１ａによって、個々のユニットブロック１６０２に分解して、ユニットブロック群１８０１ｂを生成する。

【００６６】なお、ここで、実際に機器を制御するプロセッサ構成及びその性能に関するデータ（プロセッサ構成及び性能データ）１８０１ｃ、すなわち、制御用プロセッサを何台用いて、それらをどのような形態で接続するか、ということに関する情報と、制御用プロセッサのプログラムコードの処理時間などに関する情報に基づいて、各ユニットブロック１６０２（図２７を参照）の処理時間を推定し、それが制御上必要とされる処理時間（例えば、シーケンス制御におけるスキャンタイムやロボット制御におけるサンプリング周期など）の基準を満たしているかどうかを評価する。これを行うのがユニットブロック処理時間推定及び評価手段１８０１ｄであり、これにより各ユニットブロック１６０２の処理時間に関するデータが作成される。なお、プログラムコードの処理時間については、各命令コードの処理に要する時間（制御用プロセッサ毎に予め調べられているものとする）を実行される命令コードの分だけ加算することで求めることができる。例えば、シーケンス制御プロセッサにおけるスキャンタイム（１処理に要する時間）であれば、１スキャンで実行する分だけ命令コードの処理時間を加算することにより、スキャンタイムを推定することができる。

【００６７】次に、ユニットブロック結合手段１８０１ｅは、各ユニットブロック１６０２の処理時間の推定データに基づいて、いくつかのユニットブロック１６０２を結合し、最終的に各プロセッサで実行されるプロセッサ別プログラム群１８０３を生成する。本実施例では、図２８のようなプログラムコードの実行手順をとるため、各プロセッサにおいては、その制御上必要とされる処理時間の基準を満たす範囲内で、複数のユニットブロック１６０２を実行することが可能である。すなわち、いくつかのユニットブロック１６０２の推定処理時間を加算して、それが制御上の基準の範囲内であれば、それらのユニットブロック１６０２を結合し、それをプロセッサ別プログラム１８０２とすることができる。ただし、ユニットブロック１６０２を結合するとき、特定のプロセッサに処理負荷が集中しないように、プロセッサの性能に応じた最適な負荷配分がされていることが望ましい。そこで、ユニットブロック結合手段１８０１ｅは、このように処理負荷の配分が最適になるように、ユニットブロック１８０１ｅを結合して、プロセッサ別プログラム群１８０３を生成する。すなわち、個々のプロセッサの能力をオーバーしないよう、かつ、最大時間を

要するプロセッサの処理時間が最小になるように処理負荷を配分する。

【0068】図29には、上記のプロセッサ別モジュール分割手段1801によって、機能別プログラムモジュール1601を分割して、プロセッサ別プログラム群1802を生成した例を示す。ここでは、機能別プログラムモジュール1601に含まれる3個のユニットブロック1901、1902、1903から、2個のプロセッサ別プログラム1904、1905を生成した例を示している。

【0069】既述の図10、図11、図12に示したシーケンス制御プログラムを例に取って説明すると、これらのプログラムを実行するシーケンス制御プロセッサが1個であり、これらのプログラムの全てを実行した場合のスキャンタイムが所定の時間以内であれば、これらのプログラムを一つのシーケンス制御プログラムとして結合する。これに対し、例えばシーケンス制御プロセッサが2個の場合、例えば、図10を一つのプログラム、図11と図12を結合して一つのプログラムとし、これら二つのプログラムをそれぞれのシーケンス制御プロセッサに対して出力する。

【0070】なお、上記図1に示したユニットブロック結合手段1801eにおいて生成されるプロセッサ別プログラム1802は、さらに、その出力先となるプロセッサで実行可能な形式のプログラムコードに変換する必要があることも想定される。このような場合、ユニットブロック結合手段1801eの後に、各プロセッサ用言語に変換するためのプロセッサ別のコード変換手段を予め設けることで対応することができる。

【0071】続いて、図30には、以上のようにしてセル制御プログラム102（図9を参照）から生成されたプロセッサ別プログラム群1803（図1を参照）、すなわち、シーケンス制御プロセッサ用プログラム群110、ロボット制御プロセッサ用プログラム群111、画像処理プロセッサ用プログラム群112（図2を参照）を解釈実行し、セルを制御するFAコントローラの一例を示す。なお、この図30に示されるコントローラは、シーケンス制御プロセッサ群113、ロボット制御プロセッサ群114、画像処理プロセッサ群115を共有バス2004によって結合した、いわゆる、マルチプロセッサ型コントローラ2001を構成している。また、図中の符号2014はセルを示しており、このセルは、周辺機器（I／O）2011、ロボット2012、視覚センサ2013を含んでいる。

【0072】この図の構成において、シーケンス制御プロセッサ2005は、共有バス2004とのインタフェースであるバスドライバ2005d、さらに、シーケンス制御プロセッサ用プログラムを解釈実行する演算手段2005aなどを含んで構成されており、また、I／Oインタフェース2008を介して、その制御対象である周辺機器（I／O）2011に接続されている。

【0073】ロボット制御プロセッサ2006は、バスドライバ2006f、ロボットの手先の位置姿勢を教示するための教示手段2006b、ロボット制御プロセッサ用プログラムを解釈実行する演算手段2006aなどから構成され、モータ駆動回路2009を介して、その制御対象であるロボット2012に接続される。

【0074】そして、画像処理プロセッサ2007は、バスドライバ2007f、基準画像を教示するための教示手段2007b、画像処理プロセッサ用プログラムを解釈実行する演算手段2007aなどから構成され、画像インタフェース2010を介して、工業用テレビカメラなどで構成される視覚センサ2013に接続されている。

【0075】さらに、上記のシーケンス制御プロセッサ群113、ロボット制御プロセッサ群114、画像処理プロセッサ群115で実行されるプログラムには、機器の動作タイミングに関する情報が組み込まれており、各プログラムを実行する際に、このような動作タイミングの同期を取るためには、各プロセッサの間で必要なデータを共有する必要がある。そこで、各プロセッサがアクセスできる共有バス2004には、共有データ記憶手段2003が接続されている。

【0076】図31には、上記共有データ記憶手段2003に格納される共有データ2101の構成を示す。先に述べたように、セル制御プログラム102（図9を参照）では、各ユニットの動作シーケンスを状態遷移の考え方に基づいて記述しているので、それから生成された各プログラムを実行する際にも、活性化された状態NO．を各プロセッサの間で共有することによって、ユニットの動作タイミングの同期をとることができる。そこで、共有データ2101には、ユニット別状態データ2102として、活性化された状態NO．2101aと、その実行ステータス2102b（活性化状態に関するより詳細な情報）が格納されている。各プロセッサは、それぞれのプログラムを実行中に、このユニット別状態データ2102にアクセスすることで、各プログラム中に記述された動作タイミングの同期をとることができる。また、共有データ2101には、ユニット別状態データ2102の他に、各プロセッサ間で共有すべき各種データを、共有変数2103として格納することができる。

【0077】なお、上記の図30に示したマルチプロセッサ型コントローラ2001は、通信手段2002によって、例えばパーソナルコンピュータなどのホストコンピュータ2015と接続される。このホストコンピュータ2015には、セル制御プログラム編集手段101（図2を参照）や機能別モジュール分離変換手段103、プロセッサ別モジュール分割手段1801のソフトウェアを搭載することが可能であり、ここで作成されたプログラムは、上記の通信手段2002によって各プロ

セッサに送信することができる。

【００７８】また、図３２は、上記図３０に示したマルチプロセッサ型コントローラ２００１とは異なり、従来の一般的な構成を持つＦＡシステムの制御装置の構成を示すものである。すなわち、セル２２０７を制御するために、プログラマブルコントローラ２２０１、ロボットコントローラ２２０２、画像処理装置２２０３をネットワーク２２０８で接続した例である。このように幾つかのコントローラを組み合わせた構成の場合でも、図３１に示したような共有データ２１０１を、ネットワーク２２０８を通じて、それぞれのコントローラ同士で交換し、共有することで、ユニット間の動作タイミングの同期をとることができる。よって、本発明の制御方法は、従来型の構成を持つ制御装置にも適用することが可能である。

【００７９】また、図３３には、さらに別の構成を持つＦＡシステムの制御装置を示す。これは複数のマルチプロセッサ型コントローラ２３０１、２３０２をネットワーク２３０６で接続したもので、この場合でも、各コントローラが、ネットワーク２３０６を経由して、図３１に示したような共有データ２１０１を交換することで、ユニット間の同期を取ることができ、よって、図３３のようなセル集合２３０５、つまり、より規模の大きいＦＡシステムに対しても、本発明は適用可能であることが理解される。

【００８０】さらに、上記の図３０に示したようなマルチプロセッサ型コントローラ２００１と、上記の図３２に示したような従来型のコントローラが混在した制御システムの場合でも、上記の図３３と同様にして、本発明は適用可能である。なお、上記した実施の形態の例では、自動機械としてロボットを含んだセルを対象としたが、ロボット以外の自動機械、例えばＮＣ工作機械などを含んだセルに対しても、同様にして本発明を適用することは可能である。

【００８１】また、上記の例ではプログラミング可能な周辺機器として視覚センサを含んだセルを対象としたが、視覚センサ以外のプログラミング可能な周辺機器を含んだセルに対しても、同様にして本発明を適用することは可能である。加えて、上記の実施の形態では、通常のＦＡシステムを対象としたが、さらに、複数のマニピュレータ、センサ、周辺機器などから構成される知能ロボットシステムに対しても、同様にして本発明を適用することは可能であり、それ故、本発明におけるＦＡシステムとは、かかる知能ロボットシステムをも含む概念である。

【００８２】

【発明の効果】以上の詳細な説明からも明らかなように、本発明によれば、ユーザが、ロボットなどの自動機械と視覚センサなどのプログラミング可能な周辺機器を含む複数機器を組み合わせて、ＦＡシステムを構築する際に、各機器を制御するプログラムをセル制御プログラムとして一本化して記述することができることはもちろんのこと、さらに、このセル制御プログラムから、ＦＡシステムの制御装置のプロセッサの構成と性能に応じて、各プロセッサの処理負荷の配分が最適となるように、各プロセッサ用のプログラムを自動的に生成することができるので、従来のように制御装置の構成に応じて複数のプログラムを作成したり、幾つものプログラミング言語を習得したりする必要がなくなり、その結果として、ＦＡシステムの制御プログラムの開発効率を向上することができるという効果を発揮する。

【図面の簡単な説明】

【図１】本発明の特徴点であるＦＡシステムの制御装置におけるプロセッサ別モジュール分割手段のプロセッサ別プログラム生成の処理プロセスを示す図である。

【図２】本発明の実施の形態になるＦＡシステムの制御装置における処理プロセスを模式的に示すブロック図である。

【図３】上記図２のセル制御プログラム編集手段における端末の画面表示の一例を示す図である。

【図４】上記本発明を適用した電子部品挿入作業の一部を表すペトリネットの一例を示す図である。

【図５】上記図４の電子部品挿入作業における部品挿入直前の様子を示す図である。

【図６】上記図４の電子部品挿入作業における部品挿入直後の様子を示す図である。

【図７】上記図４のペトリネットをコード化したセル制御プログラムのｃｅｌｌブロックを示す図である。

【図８】上記図４のペトリネットをコード化したセル制御プログラムのｄｅｆブロックを示す図である。

【図９】上記図２の機能別モジュール分離変換手段において、機能別プログラムモジュールを生成する処理プロセスを示すブロック図である。

【図１０】上記図７及び図８に示したセル制御プログラムから生成したシーケンス制御プログラムの一部（ＲＯＢ１のユニットブロック）を示す図である。

【図１１】上記図７及び図８に示したセル制御プログラムから生成したシーケンス制御プログラムの一部（ＲＯＢ２のユニットブロック）を示す図である。

【図１２】上記図７及び図８に示したセル制御プログラムから生成したシーケンス制御プログラムの一部（ＣＡＭ１及びＣＡＭ２のユニットブロック）を示す図である。

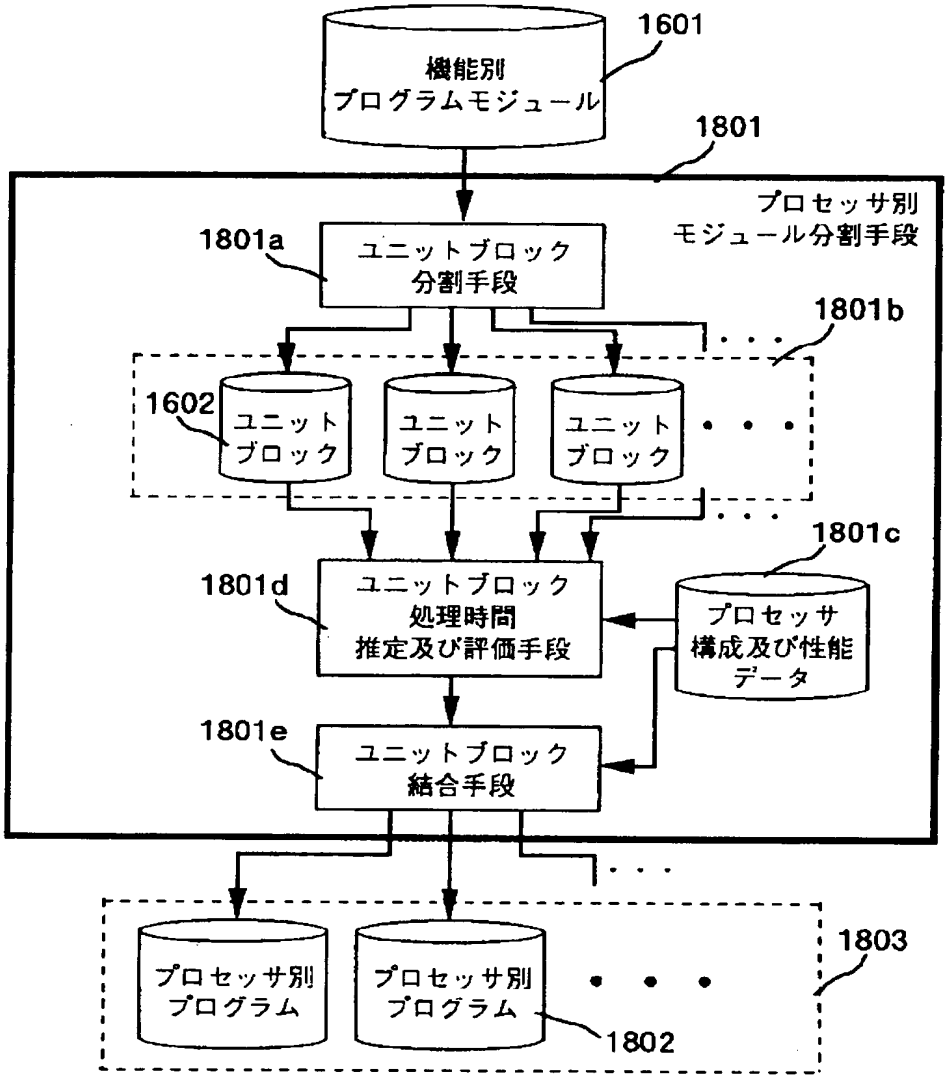
【図１３】上記図７及び図８に示したセル制御プログラムから生成したロボット制御プログラムの一例を示す図である。

【図１４】上記図７及び図８に示したセル制御プログラムから生成した画像処理プログラムの一例を示す図である。

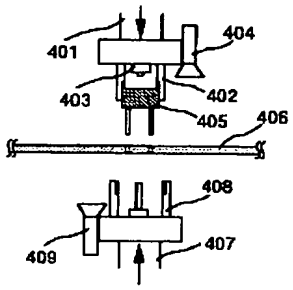
【図１５】上記図９に示す機能別モジュール分離変換手

段の処理の概略を示すフローチャートである。
【図16】上記図15に示すフローの機能別データ抽出処理の詳細を示すフローチャートである。
【図17】上記図16に示すフローの1プレース毎のネット構造解析処理の詳細を示すフローチャートの前半部分である。
【図18】上記図16に示すフローの1プレース毎のネット構造解析処理の詳細を示すフローチャートの後半部分である。
【図19】上記図16に示すフローの1プレース毎のコマンド列解析処理の詳細を示すフローチャートである。
【図20】上記図16に示すフローの1プレース毎の終了条件解析処理の詳細を示すフローチャートである。
【図21】上記図15に示すフローのシーケンス制御プログラム生成処理の詳細を示すフローチャートの前半部分である。
【図22】上記図15に示すフローシーケンス制御プログラム生成処理の詳細を示すフローチャートの後半部分である。
【図23】上記図15に示すフローのロボット制御プログラム生成処理の詳細を示すフローチャートである。
【図24】上記図15に示すフローの画像処理プログラム生成処理の詳細を示すフローチャートである。
【図25】上記図9に示した機能別モジュール分離変換手段によって、セル制御プログラムからシーケンス制御プログラムとロボット制御プログラムを生成する処理プロセスの一例を示す図である。
【図26】上記図9に示した機能別モジュール分離変換手段によって、セル制御プログラムからシーケンス制御プログラムと画像処理プログラムを生成する処理プロセスの一例を示す図である。
【図27】上記図1に示した機能別プログラムモジュールのブロック構成を示す図である。
【図28】上記図1に示した機能別プログラムモジュールのコード実行手順を示すフロー図である。
【図29】上記図2に示した機能別プログラムモジュールからプロセッサ別プログラム群を生成する処理プロセスの一例を示す図である。
【図30】本発明を適用した一実施の形態になる制御装置（マルチプロセッサ型コントローラ）の構成を示すブロック図である。
【図31】上記図30に示した制御装置における共有データの構成を示す図である。
【図32】本発明を適用した他の実施の形態になる制御装置（従来型のコントローラを組み合わせた制御システム）の構成を示す図である。
【図33】本発明を適用したさらに他の実施の形態になる制御装置（マルチプロセッサ型コントローラを複数個組み合わせた制御システム）の構成を示す図である。
【符号の説明】
101　セル制御プログラム編集手段
102　セル制御プログラム
103　機能別モジュール分離変換手段
103a　機能別データ抽出手段
103b　ネット構造
103c　状態終了条件
103d　I／O制御命令列
103e　ロボット制御命令列
103f　ロボット動作タイミング
103g　画像処理命令列
103h　画像処理タイミング
103i　シーケンス制御プログラム生成手段
103j　ロボット制御プログラム生成手段
103k　画像処理プログラム生成手段
104　シーケンス制御プログラムモジュール
105　ロボット制御プログラムモジュール
106　画像処理プログラムモジュール
107　シーケンス制御プロセッサ別モジュール分割手段
108　ロボット制御プロセッサ別モジュール分割手段
109　画像処理プロセッサ別モジュール分割手段
110　シーケンス制御プロセッサ用プログラム群
111　ロボット制御プロセッサ用プログラム群
112　画像処理プロセッサ用プログラム群
113　シーケンス制御プロセッサ群
114　ロボット制御プロセッサ群
115　画像処理プロセッサ群
116　セル
116a、116b　ロボット
116c、116d　視覚センサ
116e、116f　周辺機器
1601　機能別プログラムモジュール
1801　プロセッサ別モジュール分割手段
1801a　ユニットブロック分割手段
1801b　ユニットブロック群
1801c　プロセッサ構成及び性能データ
1801d　ユニットブロック処理時間推定及び結合手段
1801e　ユニットブロック結合手段
1802　プロセッサ別プログラム
1803　プロセッサ別プログラム群

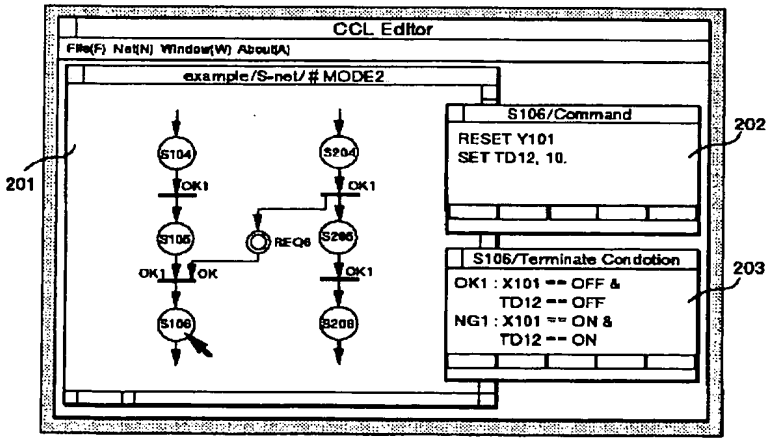
【図1】


1601
機能別
プログラムモジュール
1801
プロセッサ別
モジュール分割手段
1801a
ユニットブロック
分割手段
1801b
1602
ユニット
ブロック
ユニット
ブロック
ユニット
ブロック
1801c
1801d
ユニットブロック
処理時間
推定及び評価手段
プロセッサ
構成及び性能
データ
1801e
ユニットブロック
結合手段
1803
プロセッサ別
プログラム
プロセッサ別
プログラム
1802

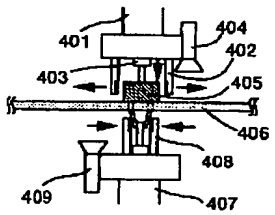
【図5】


401
404
403
402
405
406
408
409
407

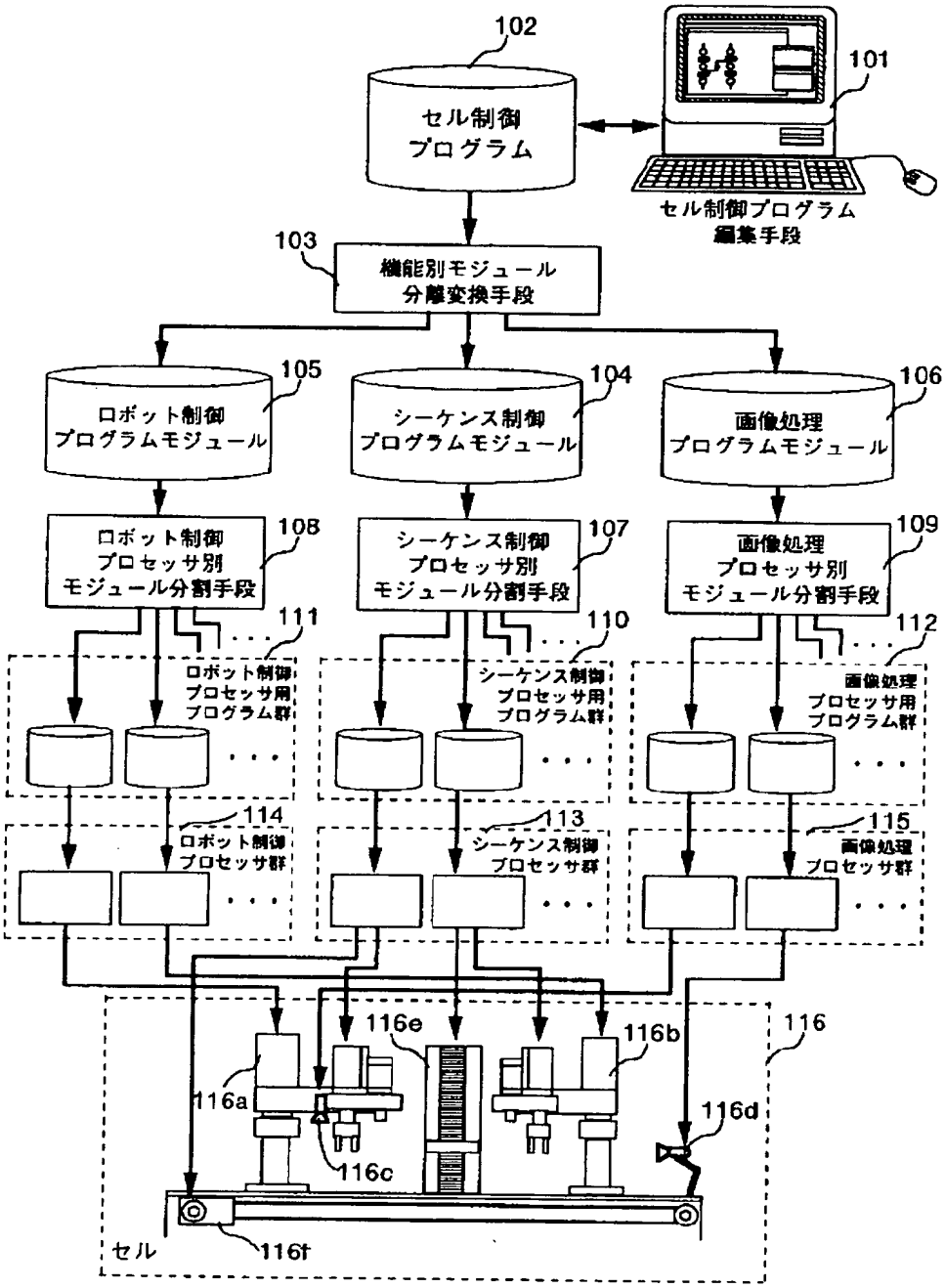
【図3】


CCL Editor
File(F) Net(N) Window(W) About(A)
example/S-net/#MODE2
S104
S204
OK1
OK1
S105
REQ6
S205
OK1
OK
OK1
S106
S206
201
S106/Command
RESET Y101
SET TD12, 10.
202
S106/Terminate Condotion
OK1 : X101 == OFF &
TD12 == OFF
NG1 : X101 == ON &
TD12 == ON
203

【図6】


401
404
402
403
405
406
408
409
407

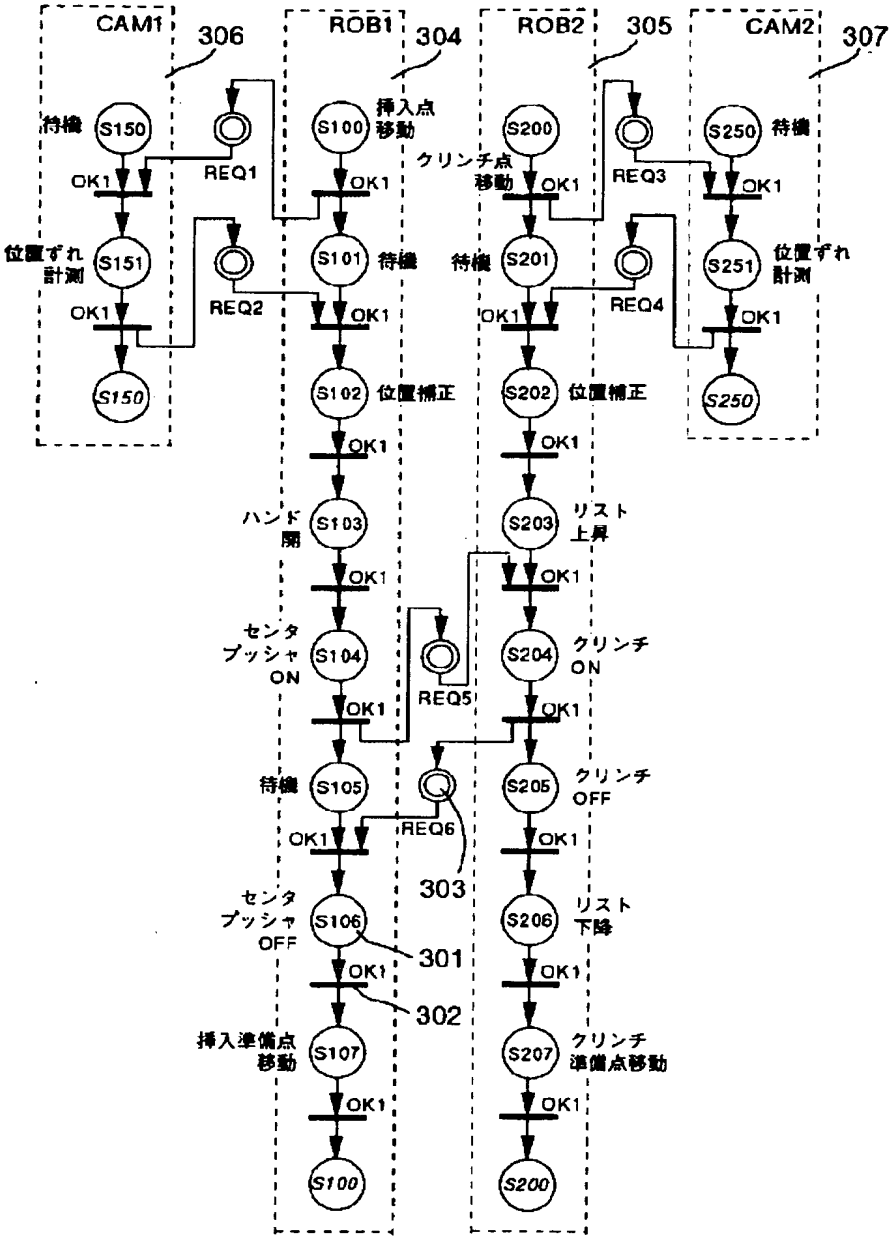
【図2】



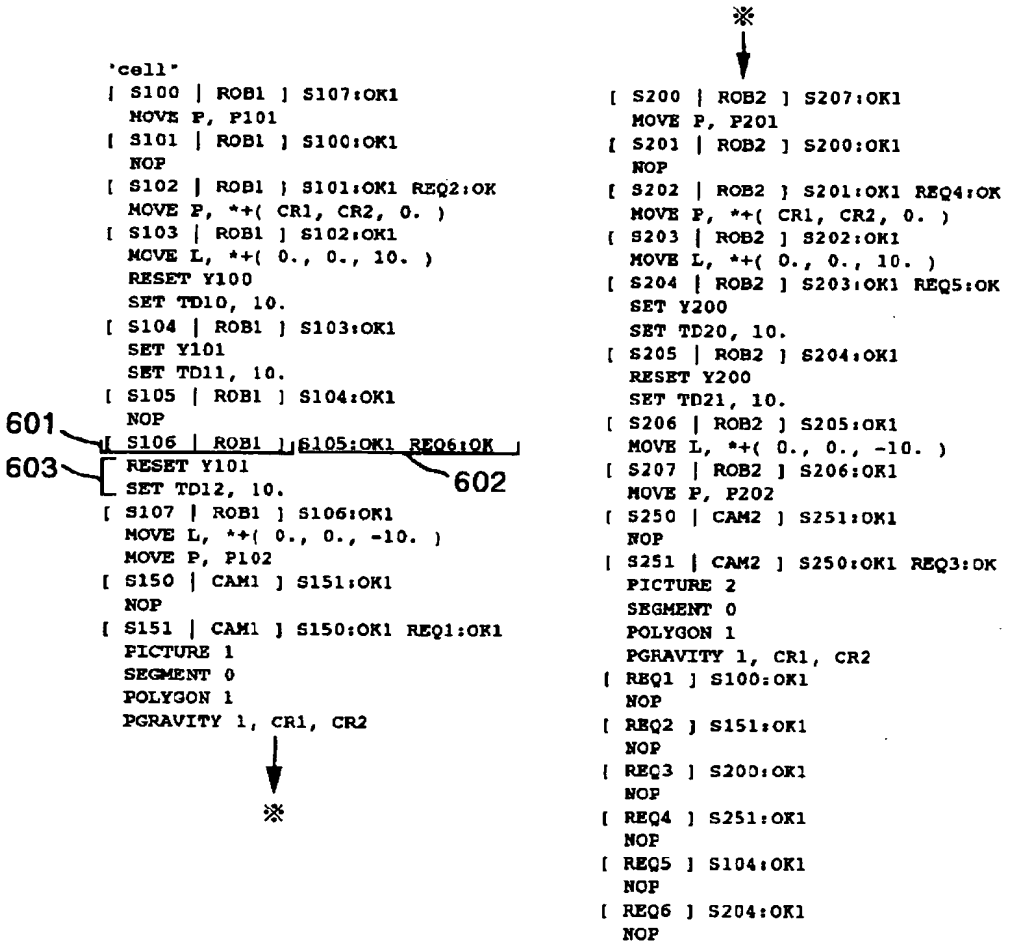
【図12】

```
#CAM1

IF S150 == CompleteNormally
&  REQ1 == CompleteNormally
->
   TERM S150
   TERM REQ1
   EXEC S151

IF S151 == CompleteNormally
->
   TERM S151
   EXEC S150
   EXEC REQ2

#CAM2

IF S250 == CompleteNormally
&  REQ3 == CompleteNormally
->
   TERM S250
   TERM REQ3
   EXEC S251

IF S251 == CompleteNormally
->
   TERM S251
   EXEC S250
   EXEC REQ4
```

【図4】


CAM1
306
ROB1
304
ROB2
305
CAM2
307
待機
S150
OK1
REQ1
挿入点
移動
S100
クリンチ点
移動
S200
REQ3
S250
待機
位置ずれ
計測
S151
REQ2
S101
待機
S201
REQ4
S251
位置ずれ
計測
S150
S102
位置補正
S202
位置補正
S250
ハンド
開
S103
S203
リスト
上昇
センタ
プッシャ
ON
S104
REQ5
S204
クリンチ
ON
待機
S105
REQ6
S205
クリンチ
OFF
303
センタ
プッシャ
OFF
S106
301
S206
リスト
下降
302
挿入準備点
移動
S107
S207
クリンチ
準備点移動
S100
S200

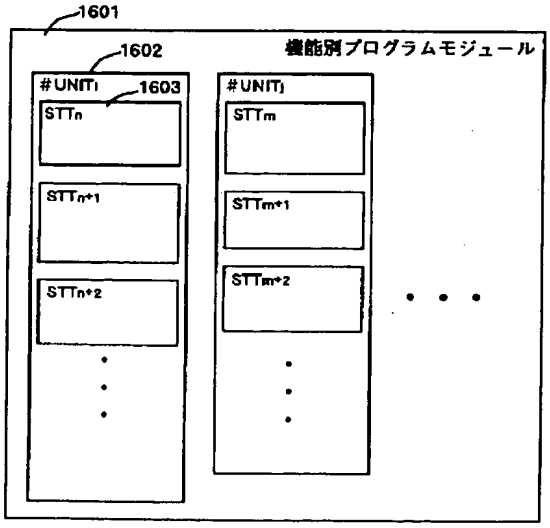
【図7】



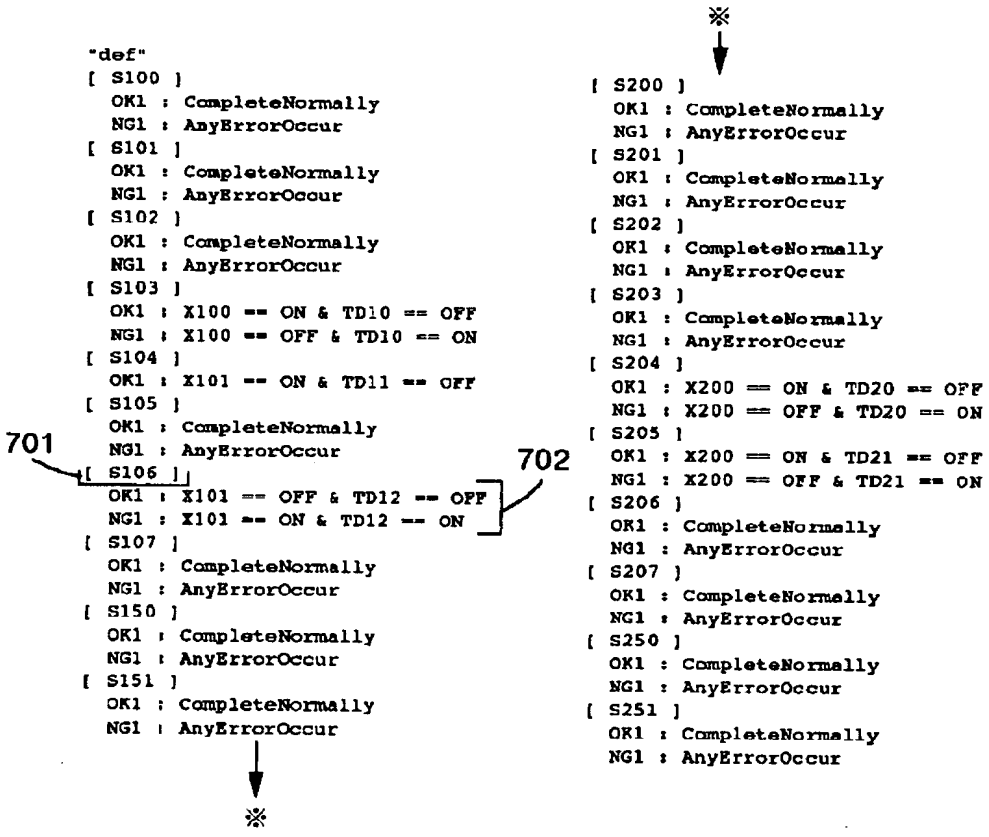
【図14】

```
#CAM1
  STT150
    NOP
  STT151
    PICTURE 1
    SEGMENT 0
    POLYGON 1
    PGRAVITY 1, CR1, CR2

#CAM2
  STT250
    NOP
  STT251
    PICTURE 2
    SEGMENT 0
    POLYGON 1
    PGRAVITY 1, CR1, CR2
```

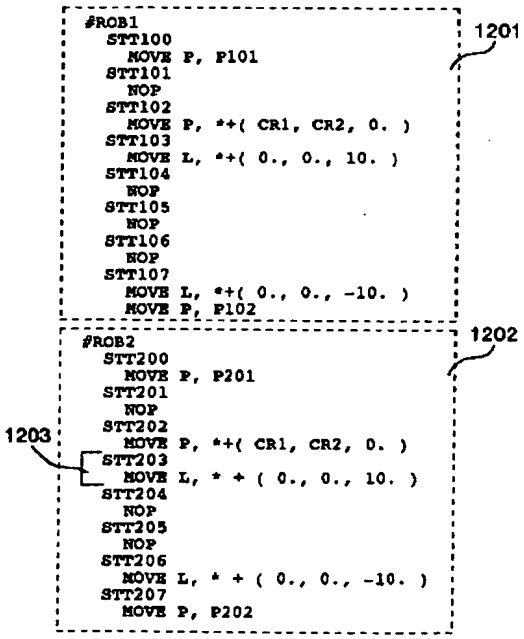
【図27】



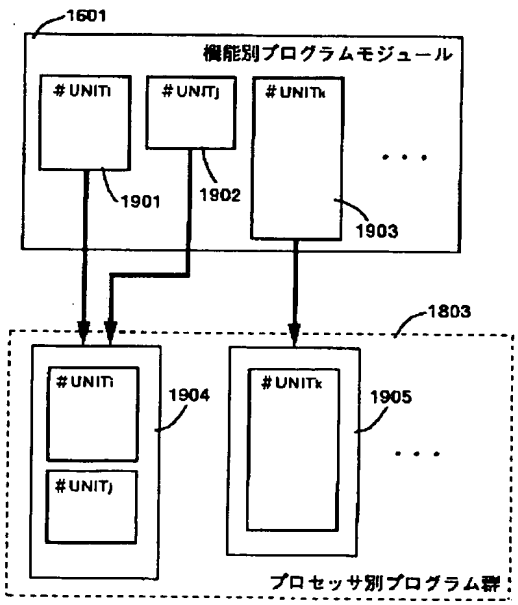
【図8】


"def"
[S100]
OK1 : CompleteNormally
NG1 : AnyErrorOccur
[S101]
OK1 : CompleteNormally
NG1 : AnyErrorOccur
[S102]
OK1 : CompleteNormally
NG1 : AnyErrorOccur
[S103]
OK1 : X100 == ON & TD10 == OFF
NG1 : X100 == OFF & TD10 == ON
[S104]
OK1 : X101 == ON & TD11 == OFF
[S105]
OK1 : CompleteNormally
NG1 : AnyErrorOccur
[S106]
OK1 : X101 == OFF & TD12 == OFF
NG1 : X101 == ON & TD12 == ON
[S107]
OK1 : CompleteNormally
NG1 : AnyErrorOccur
[S150]
OK1 : CompleteNormally
NG1 : AnyErrorOccur
[S151]
OK1 : CompleteNormally
NG1 : AnyErrorOccur
701
702
※
※
[S200]
OK1 : CompleteNormally
NG1 : AnyErrorOccur
[S201]
OK1 : CompleteNormally
NG1 : AnyErrorOccur
[S202]
OK1 : CompleteNormally
NG1 : AnyErrorOccur
[S203]
OK1 : CompleteNormally
NG1 : AnyErrorOccur
[S204]
OK1 : X200 == ON & TD20 == OFF
NG1 : X200 == OFF & TD20 == ON
[S205]
OK1 : X200 == ON & TD21 == OFF
NG1 : X200 == OFF & TD21 == ON
[S206]
OK1 : CompleteNormally
NG1 : AnyErrorOccur
[S207]
OK1 : CompleteNormally
NG1 : AnyErrorOccur
[S250]
OK1 : CompleteNormally
NG1 : AnyErrorOccur
[S251]
OK1 : CompleteNormally
NG1 : AnyErrorOccur

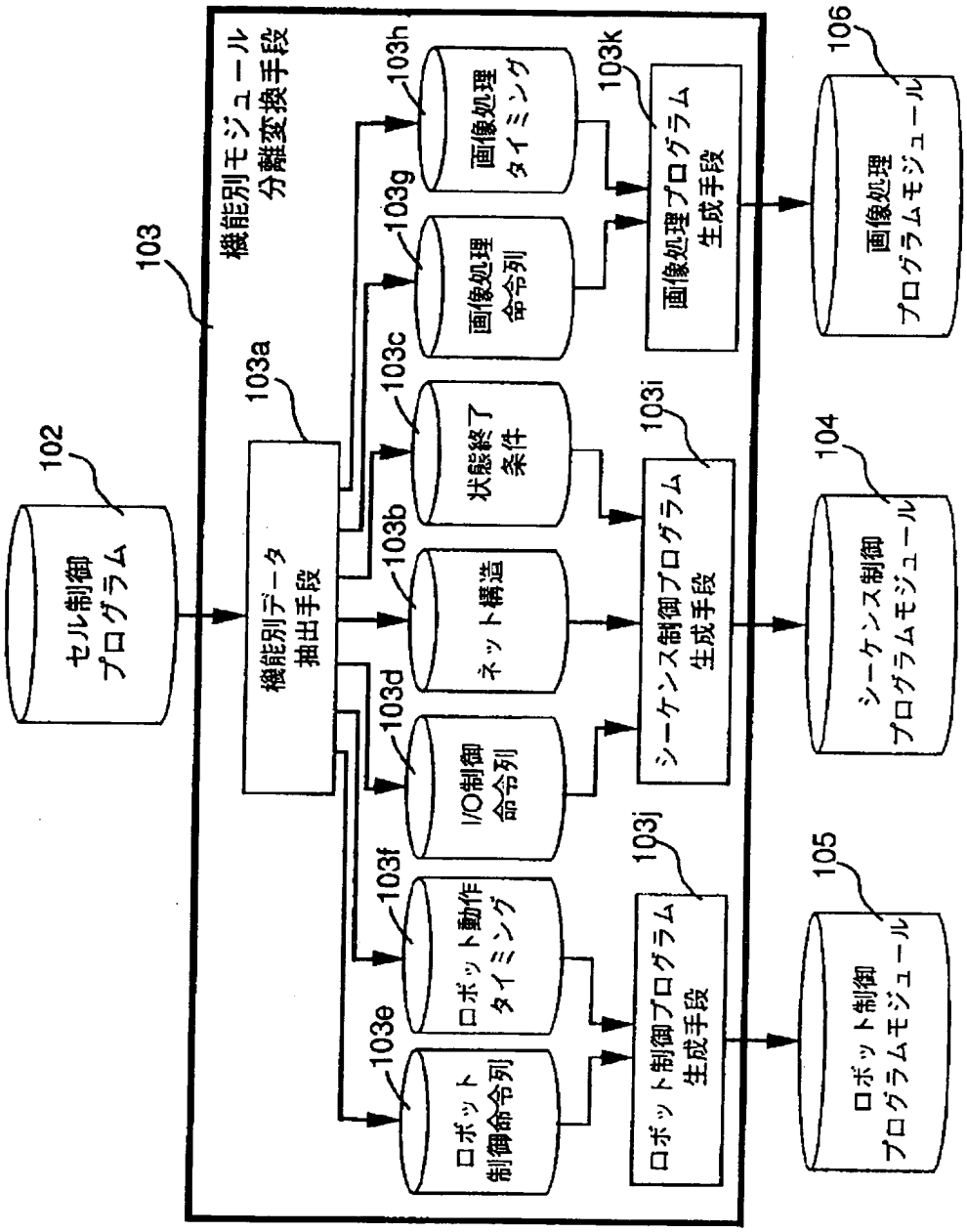
【図13】


#ROB1
STT100
MOVE P, P101
STT101
NOP
STT102
MOVE P, *+(CR1, CR2, 0.)
STT103
MOVE L, *+(0., 0., 10.)
STT104
NOP
STT105
NOP
STT106
NOP
STT107
MOVE L, *+(0., 0., -10.)
MOVE P, P102
1201
#ROB2
STT200
MOVE P, P201
STT201
NOP
STT202
MOVE P, *+(CR1, CR2, 0.)
STT203
MOVE L, * + (0., 0., 10.)
STT204
NOP
STT205
NOP
STT206
MOVE L, * + (0., 0., -10.)
STT207
MOVE P, P202
1202
1203

【図29】


1601
機能別プログラムモジュール
#UNITi
#UNITj
#UNITk
1901
1902
1903
1803
#UNITi
#UNITj
1904
#UNITk
1905
プロセッサ別プログラム群

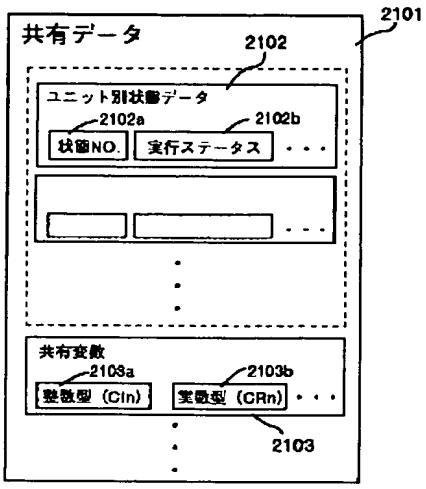
【図9】


機能別モジュール
分離変換手段
103
102
セル制御
プログラム
103a
機能別データ
抽出手段
103b
ネット構造
103c
状態終了
条件
103d
I/O制御
命令列
103e
ロボット
制御命令列
103f
ロボット動作
タイミング
103g
画像処理
命令列
103h
画像処理
タイミング
103i
シーケンス制御プログラム
生成手段
103j
ロボット制御プログラム
生成手段
103k
画像処理プログラム
生成手段
104
シーケンス制御
プログラムモジュール
105
ロボット制御
プログラムモジュール
106
画像処理
プログラムモジュール

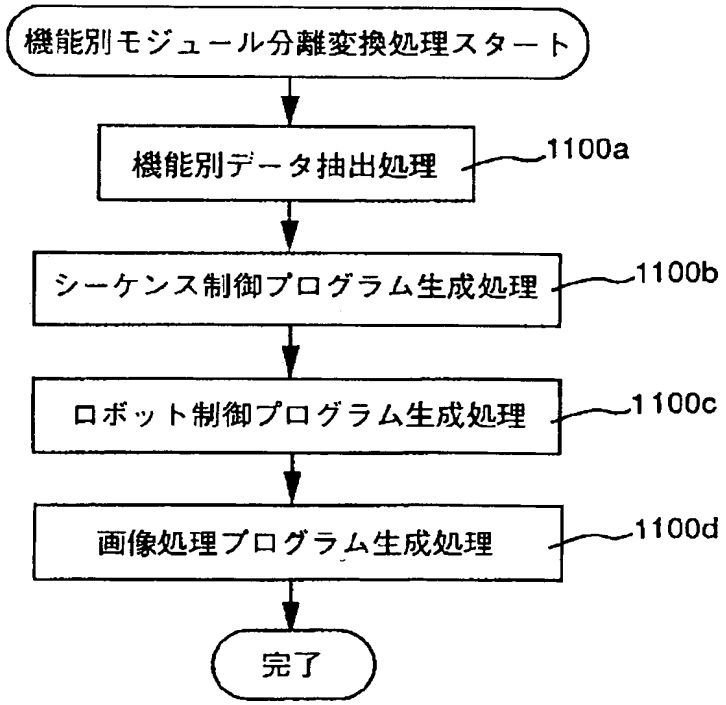
【図10】

```
#ROB1

IF S100 == CompleteNormally
->
   TERM S100
   EXEC S101
   EXEC REQ1

IF S101 == CompleteNormally
&  REQ2 == CompleteNormally
->
   TERM S101
   TERM REQ6
   EXEC S102

IF S102 == CompleteNormally
->
   TERM S102
   EXEC S103
   RESET Y100
   SET TD10 , 10.

IF S103 == Executing
&  X100 == ON
&  TD10 == OFF
->
   TERM S103
   EXEC S104
   SET Y101
   SET TD11 , 10.
        ⋮
        ※

        ※
        ⋮
IF S104 == Executing
&  X101 == ON
&  TD11 == OFF
->
   TERM S104
   EXEC S105
   EXEC REQ5

IF S105 == CompleteNormally
&  REQ6 == CompleteNormally
->
   TERM S105
   TERM REQ6
   EXEC S106
   RESET Y101
   SET TD12 , 10.

IF S106 == Executing
&  X101 == OFF
&  TD12 == OFF
->
   TERM S106
   EXEC S107

IF S107 == CompleteNormally
->
   TERM S107
   EXEC S100
```

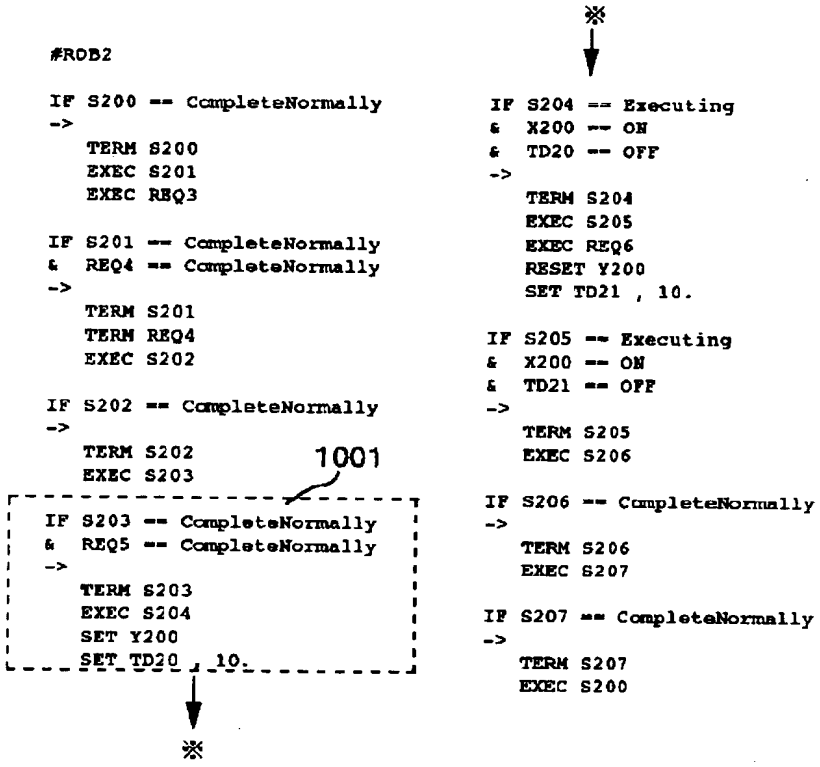
【図31】



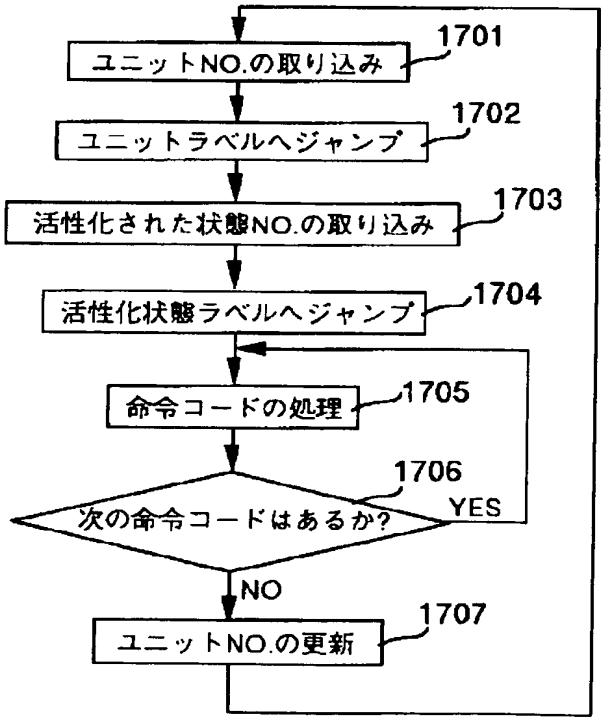
【図15】



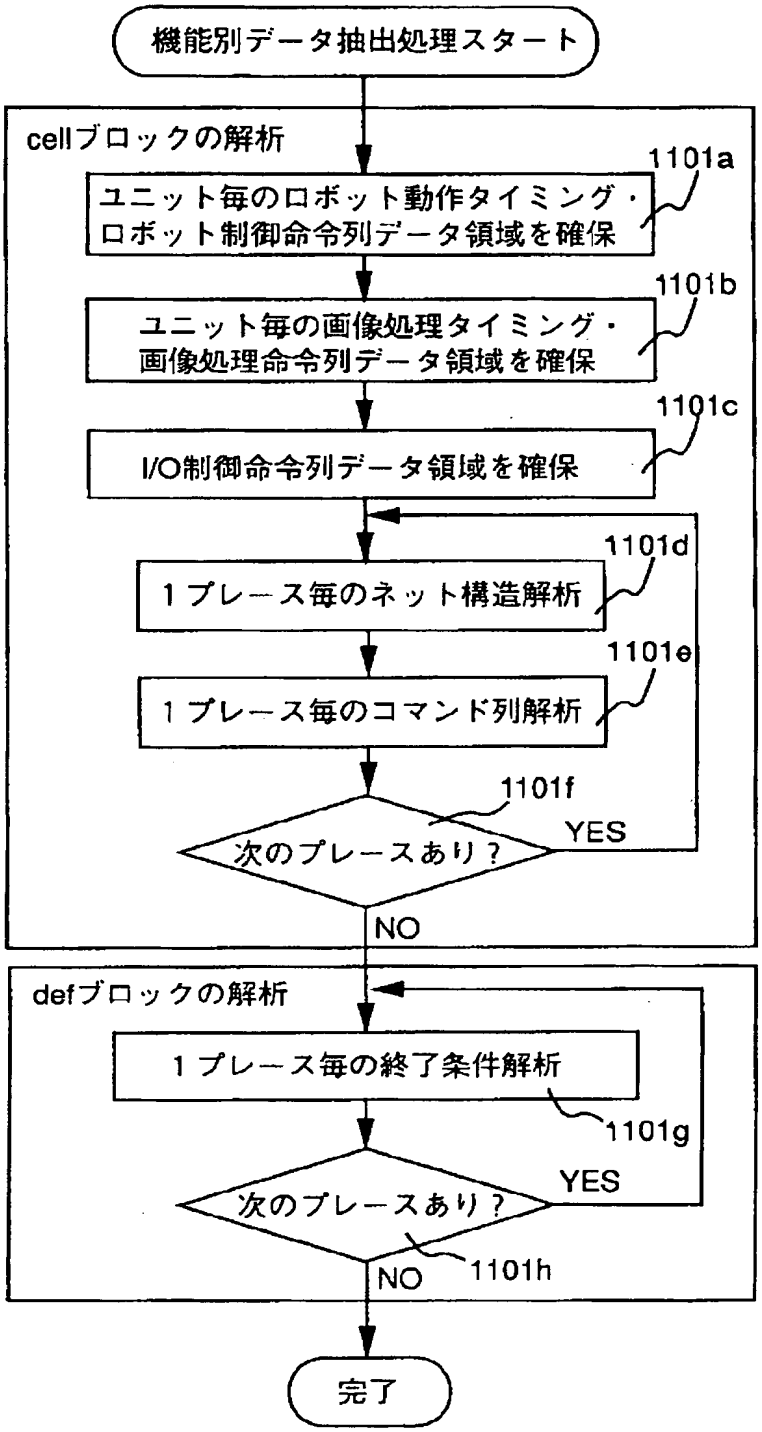
【図11】


※
#ROB2
IF S200 == CompleteNormally
->
TERM S200
EXEC S201
EXEC REQ3
IF S201 == CompleteNormally
& REQ4 == CompleteNormally
->
TERM S201
TERM REQ4
EXEC S202
IF S202 == CompleteNormally
->
TERM S202
EXEC S203
1001
IF S203 == CompleteNormally
& REQ5 == CompleteNormally
->
TERM S203
EXEC S204
SET Y200
SET TD20 , 10.
※
IF S204 == Executing
& X200 == ON
& TD20 == OFF
->
TERM S204
EXEC S205
EXEC REQ6
RESET Y200
SET TD21 , 10.
IF S205 == Executing
& X200 == ON
& TD21 == OFF
->
TERM S205
EXEC S206
IF S206 == CompleteNormally
->
TERM S206
EXEC S207
IF S207 == CompleteNormally
->
TERM S207
EXEC S200

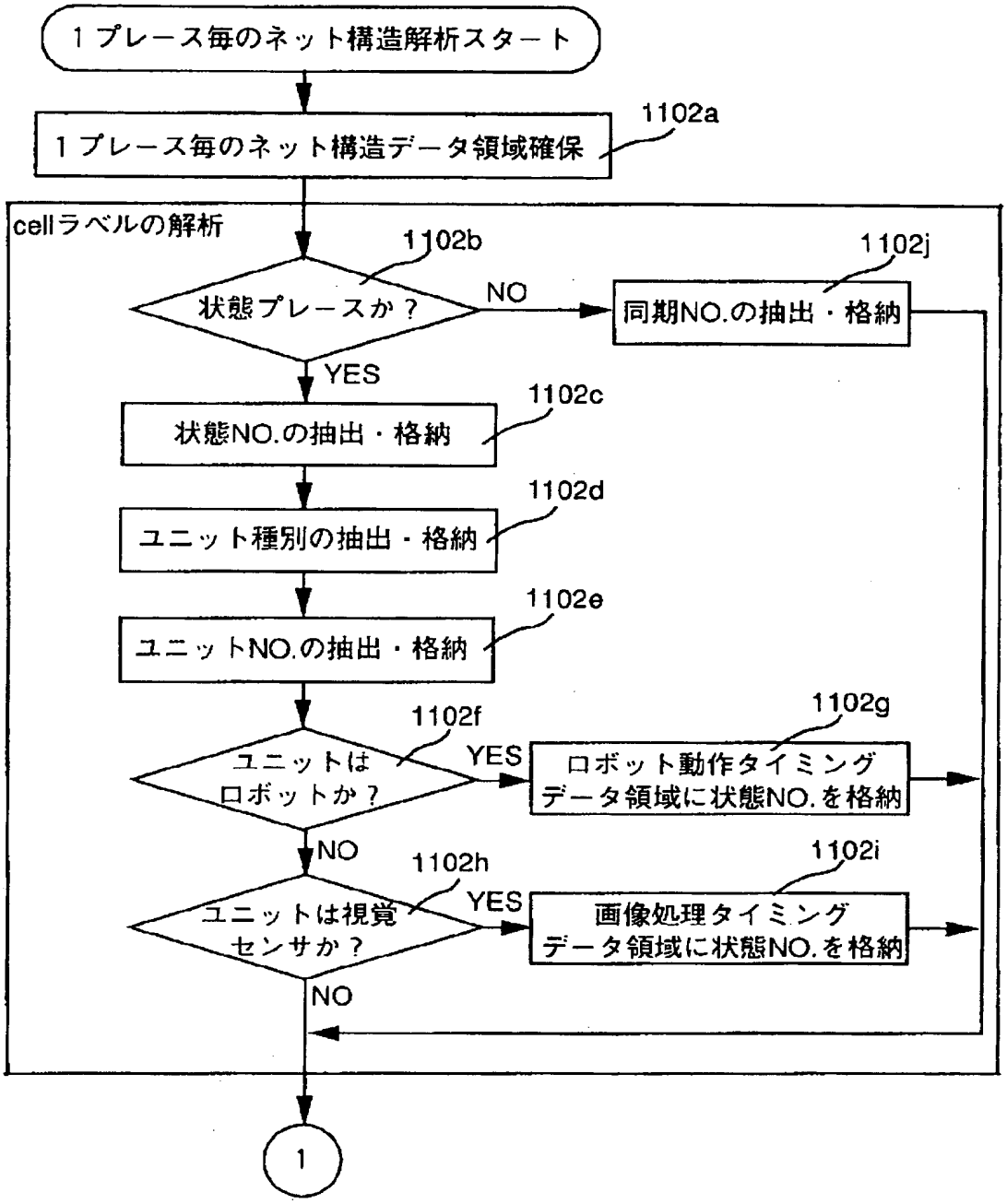
【図28】


ユニットNO.の取り込み
1701
ユニットラベルへジャンプ
1702
活性化された状態NO.の取り込み
1703
活性化状態ラベルへジャンプ
1704
命令コードの処理
1705
次の命令コードはあるか?
1706
YES
NO
ユニットNO.の更新
1707

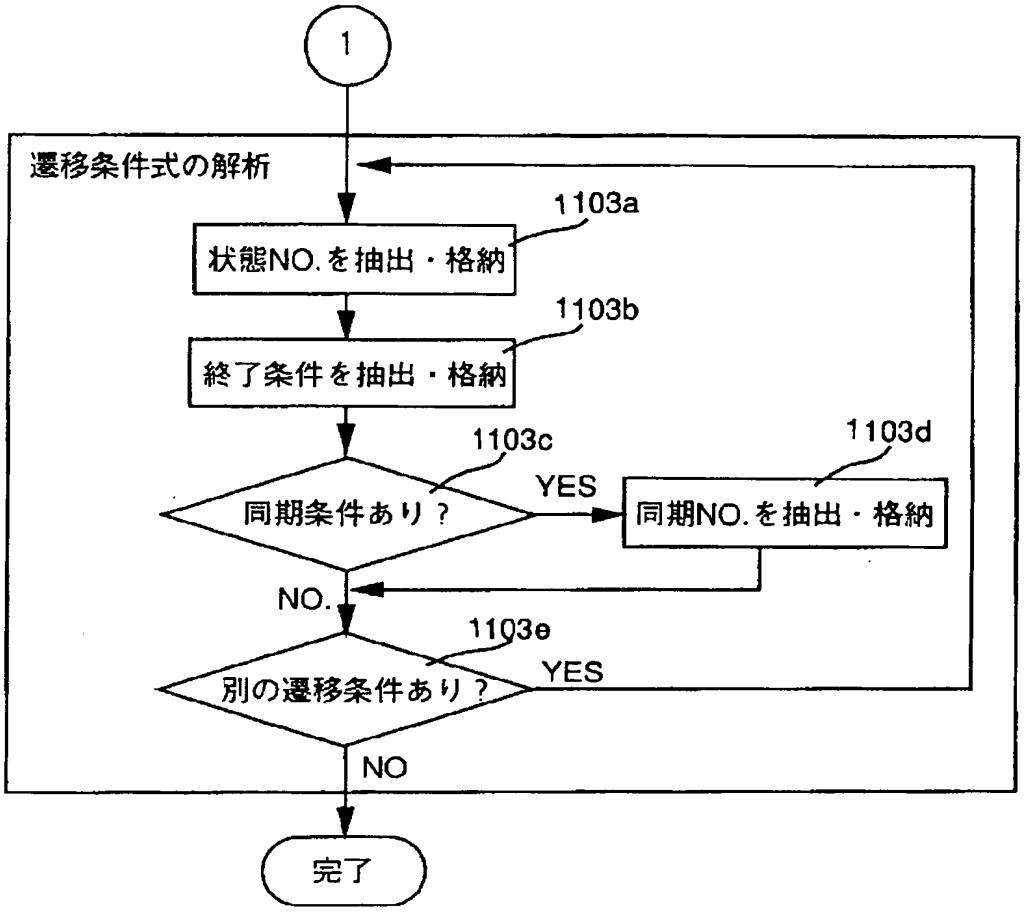
【図16】


機能別データ抽出処理スタート
cellブロックの解析
1101a
ユニット毎のロボット動作タイミング・
ロボット制御命令列データ領域を確保
1101b
ユニット毎の画像処理タイミング・
画像処理命令列データ領域を確保
1101c
I/O制御命令列データ領域を確保
1101d
1プレース毎のネット構造解析
1101e
1プレース毎のコマンド列解析
1101f
次のプレースあり？
YES
NO
defブロックの解析
1プレース毎の終了条件解析
1101g
次のプレースあり？
YES
NO
1101h
完了

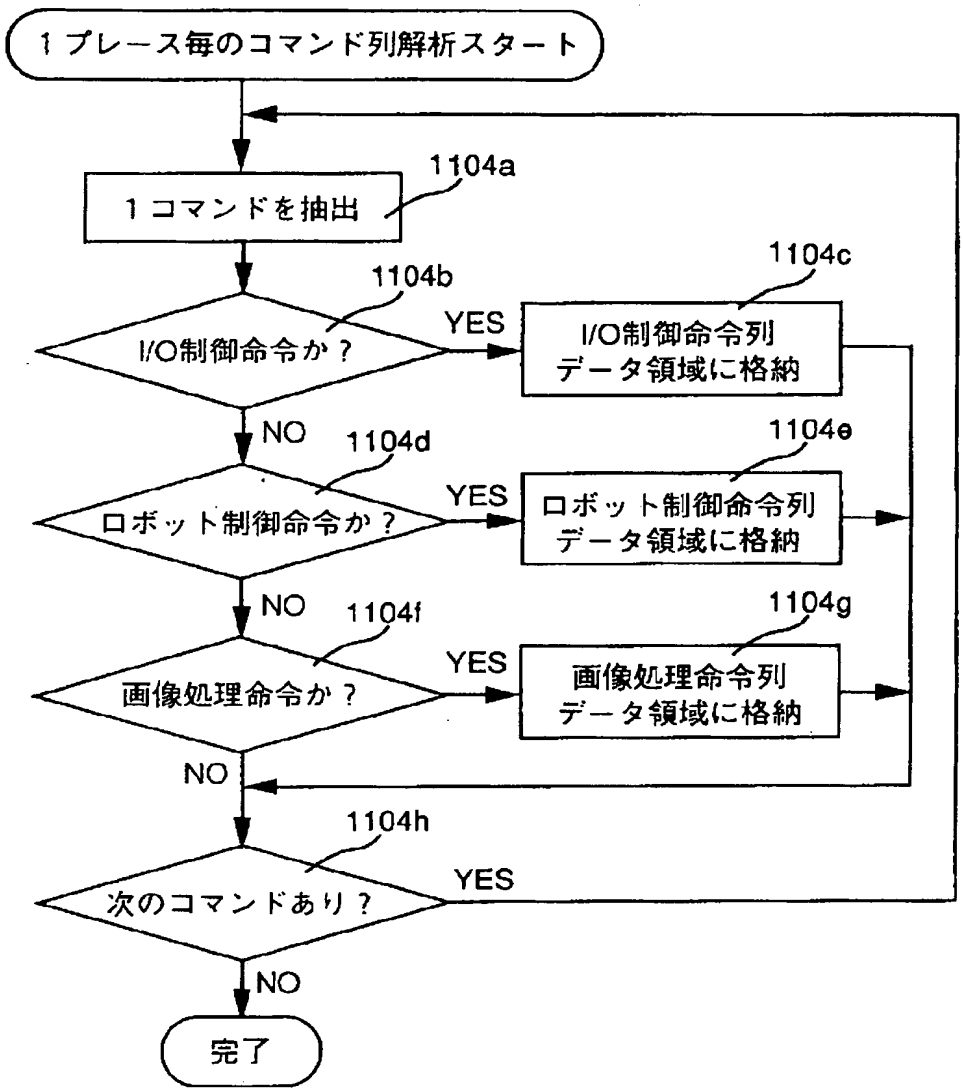
【図17】


1プレース毎のネット構造解析スタート
1プレース毎のネット構造データ領域確保
1102a
cellラベルの解析
1102b
状態プレースか？
NO
1102j
同期NO.の抽出・格納
YES
1102c
状態NO.の抽出・格納
1102d
ユニット種別の抽出・格納
1102e
ユニットNO.の抽出・格納
1102f
ユニットは
ロボットか？
YES
1102g
ロボット動作タイミング
データ領域に状態NO.を格納
NO
1102h
ユニットは視覚
センサか？
YES
1102i
画像処理タイミング
データ領域に状態NO.を格納
NO
1

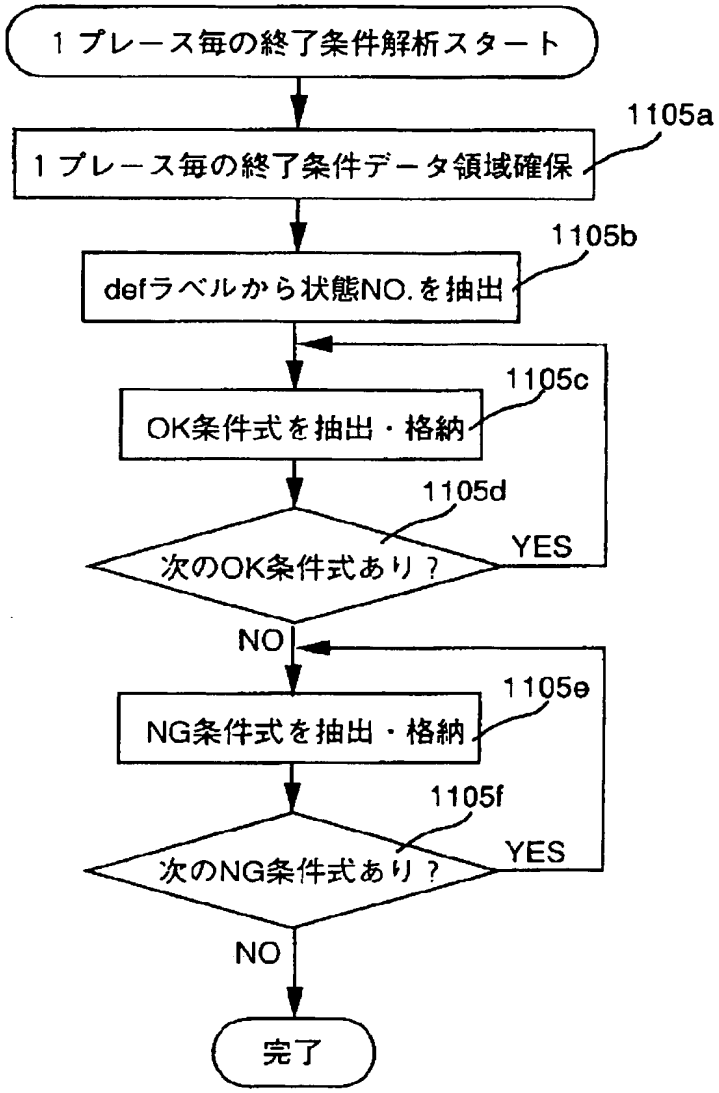
【図18】


1
遷移条件式の解析
1103a
状態NO.を抽出・格納
1103b
終了条件を抽出・格納
1103c
同期条件あり？
YES
1103d
同期NO.を抽出・格納
NO.
1103e
別の遷移条件あり？
YES
NO
完了

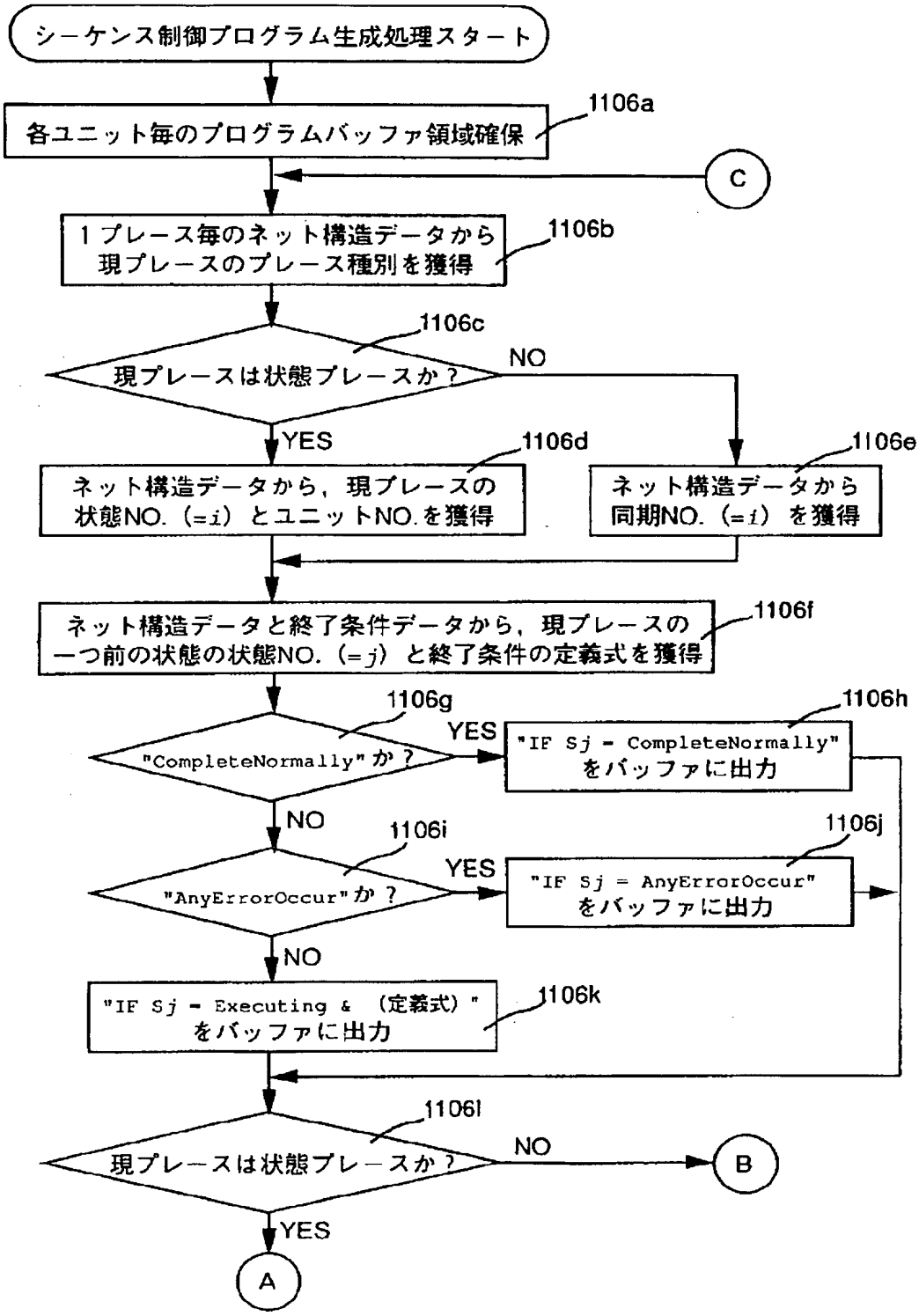
【図19】


1プレース毎のコマンド列解析スタート
1104a
1コマンドを抽出
1104b
I/O制御命令か？
YES
1104c
I/O制御命令列
データ領域に格納
NO
1104d
ロボット制御命令か？
YES
1104e
ロボット制御命令列
データ領域に格納
NO
1104f
画像処理命令か？
YES
1104g
画像処理命令列
データ領域に格納
NO
1104h
次のコマンドあり？
YES
NO
完了

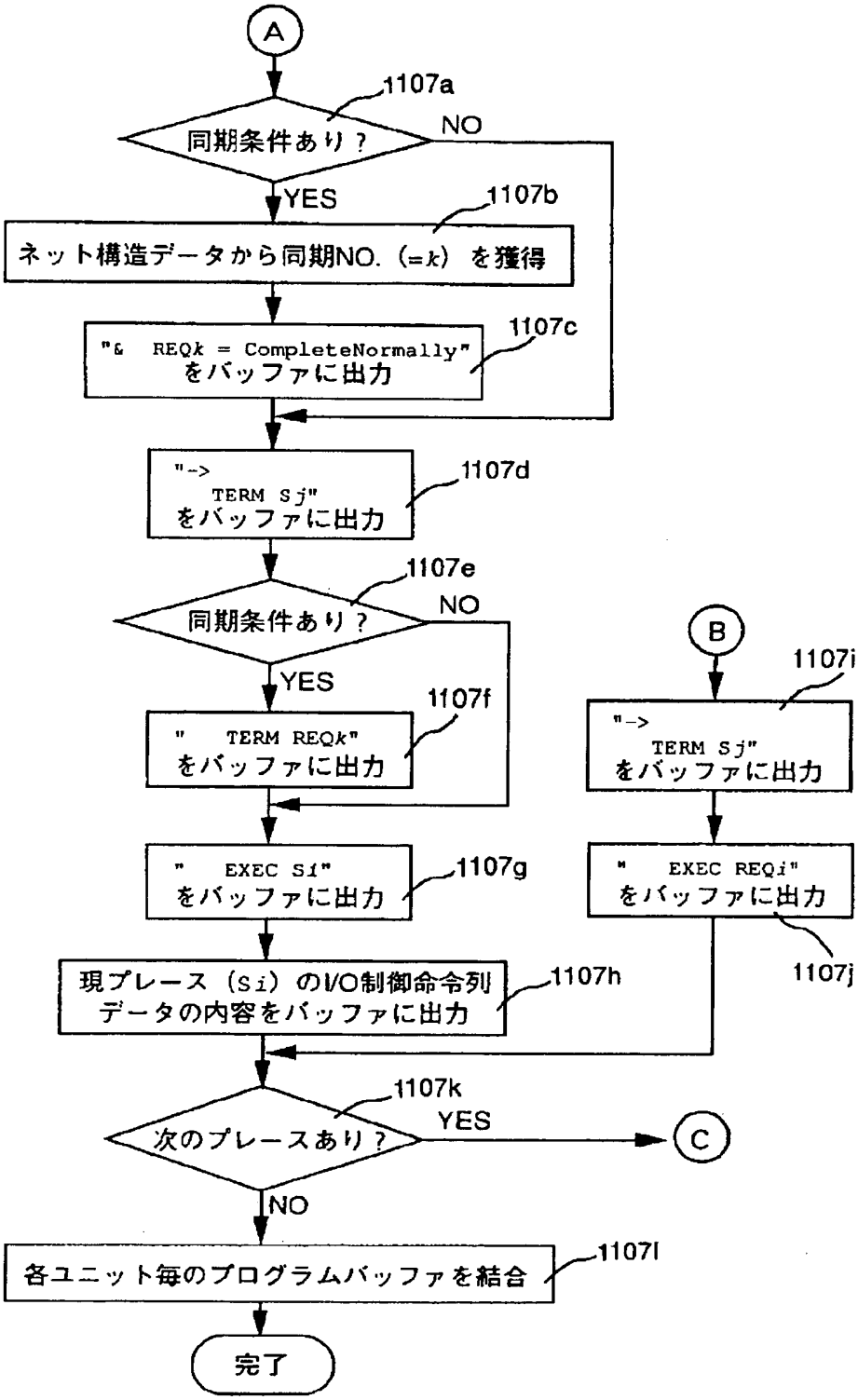
【図20】


1プレース毎の終了条件解析スタート
1105a
1プレース毎の終了条件データ領域確保
1105b
defラベルから状態NO.を抽出
1105c
OK条件式を抽出・格納
1105d
次のOK条件式あり？
YES
NO
1105e
NG条件式を抽出・格納
1105f
次のNG条件式あり？
YES
NO
完了

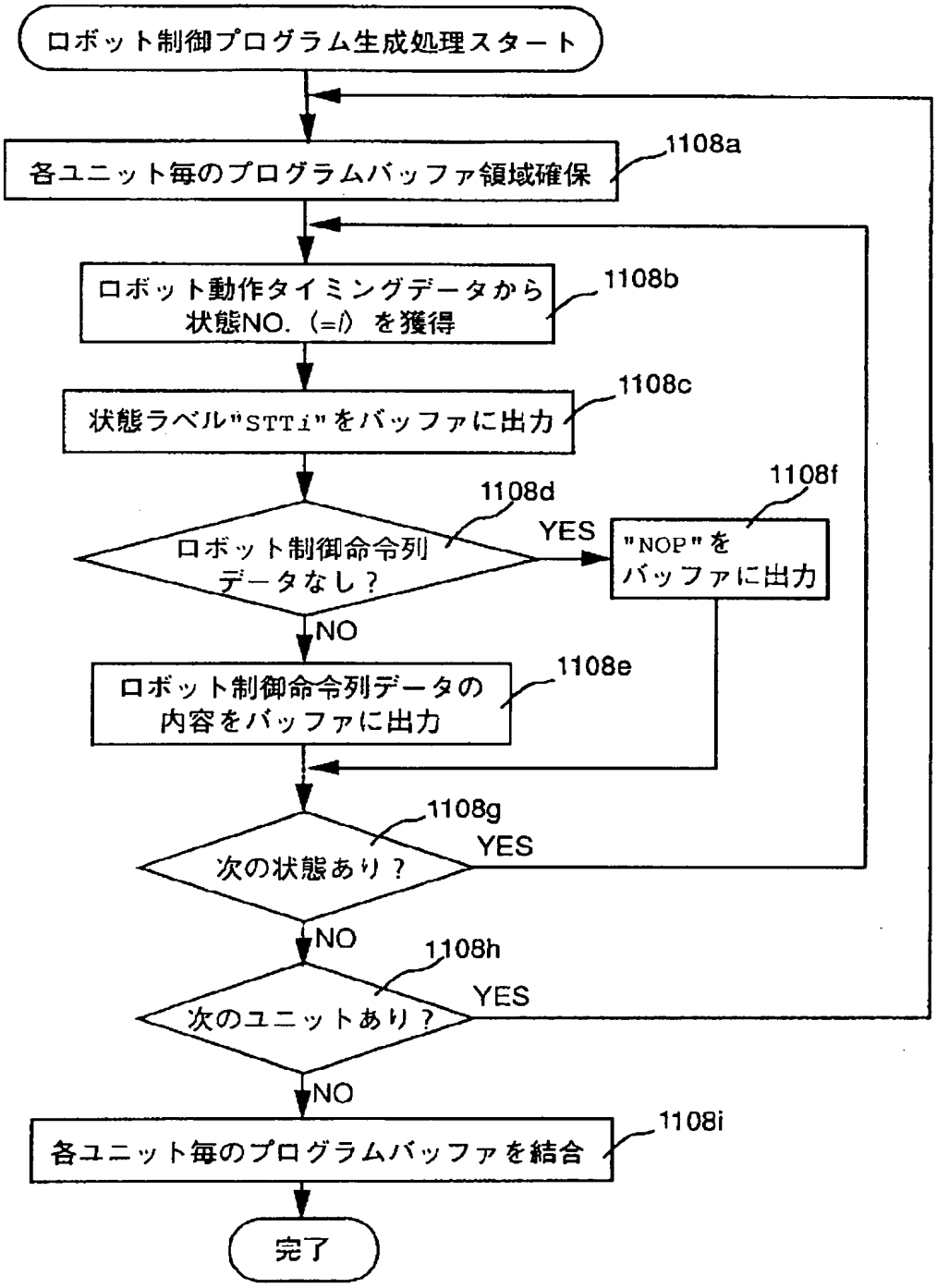
【図21】


シーケンス制御プログラム生成処理スタート
各ユニット毎のプログラムバッファ領域確保
1106a
C
1プレース毎のネット構造データから
現プレースのプレース種別を獲得
1106b
1106c
現プレースは状態プレースか？
NO
YES
1106d
ネット構造データから，現プレースの
状態NO.（=i）とユニットNO.を獲得
1106e
ネット構造データから
同期NO.（=i）を獲得
ネット構造データと終了条件データから，現プレースの
一つ前の状態の状態NO.（=j）と終了条件の定義式を獲得
1106f
1106g
"CompleteNormally"か？
YES
1106h
"IF Sj = CompleteNormally"
をバッファに出力
NO
1106i
"AnyErrorOccur"か？
YES
1106j
"IF Sj = AnyErrorOccur"
をバッファに出力
NO
"IF Sj = Executing & （定義式）"
をバッファに出力
1106k
1106l
現プレースは状態プレースか？
NO
B
YES
A

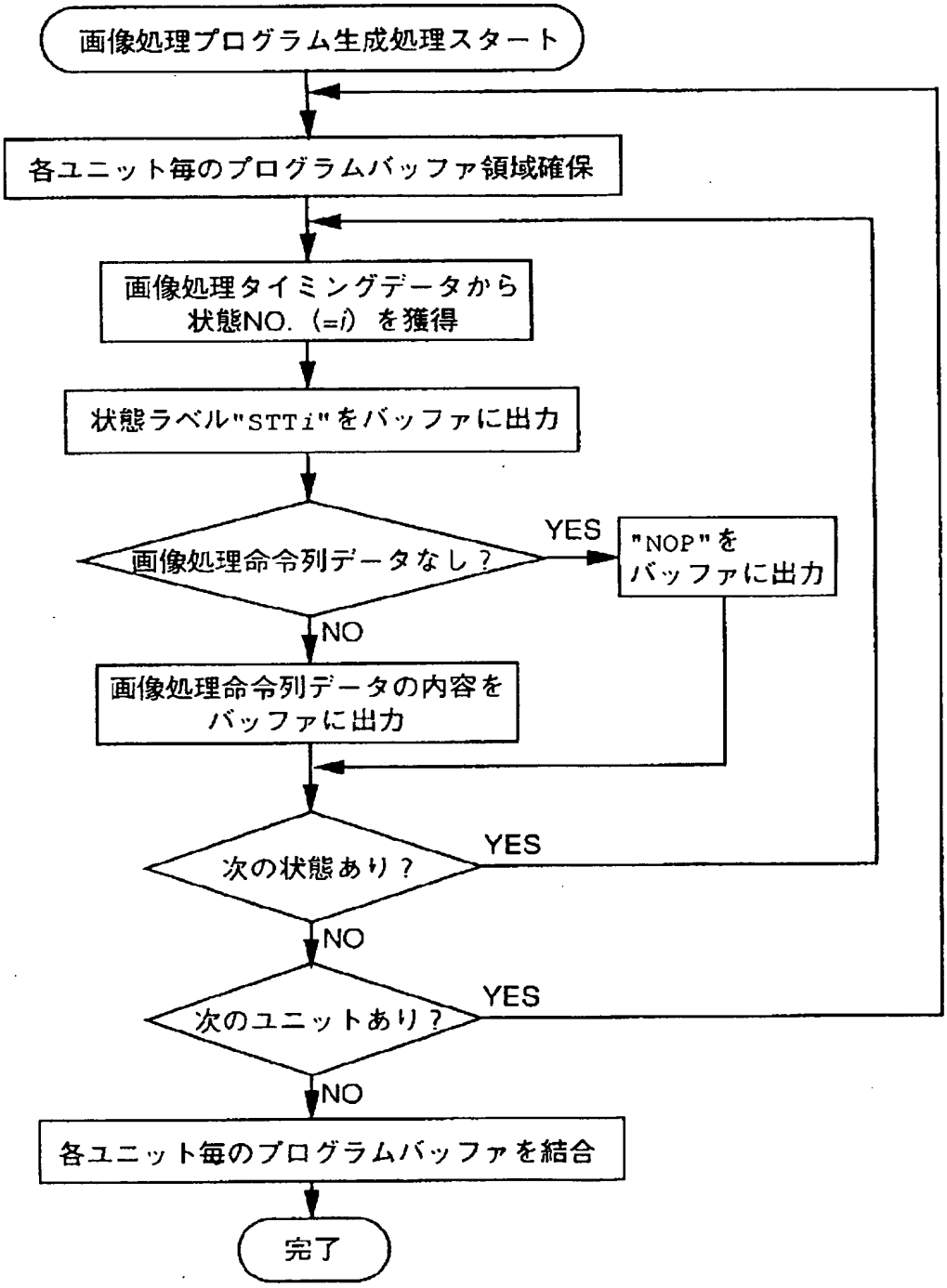
【図22】


A
1107a
同期条件あり？
NO
YES
1107b
ネット構造データから同期NO.（=k）を獲得
1107c
"& REQk = CompleteNormally"
をバッファに出力
"->
TERM Sj"
をバッファに出力
1107d
1107e
同期条件あり？
NO
YES
1107f
" TERM REQk"
をバッファに出力
" EXEC Si"
をバッファに出力
1107g
現プレース（Si）のI/O制御命令列
データの内容をバッファに出力
1107h
B
1107i
"->
TERM Sj"
をバッファに出力
" EXEC REQi"
をバッファに出力
1107j
1107k
次のプレースあり？
YES
C
NO
各ユニット毎のプログラムバッファを結合
1107l
完了

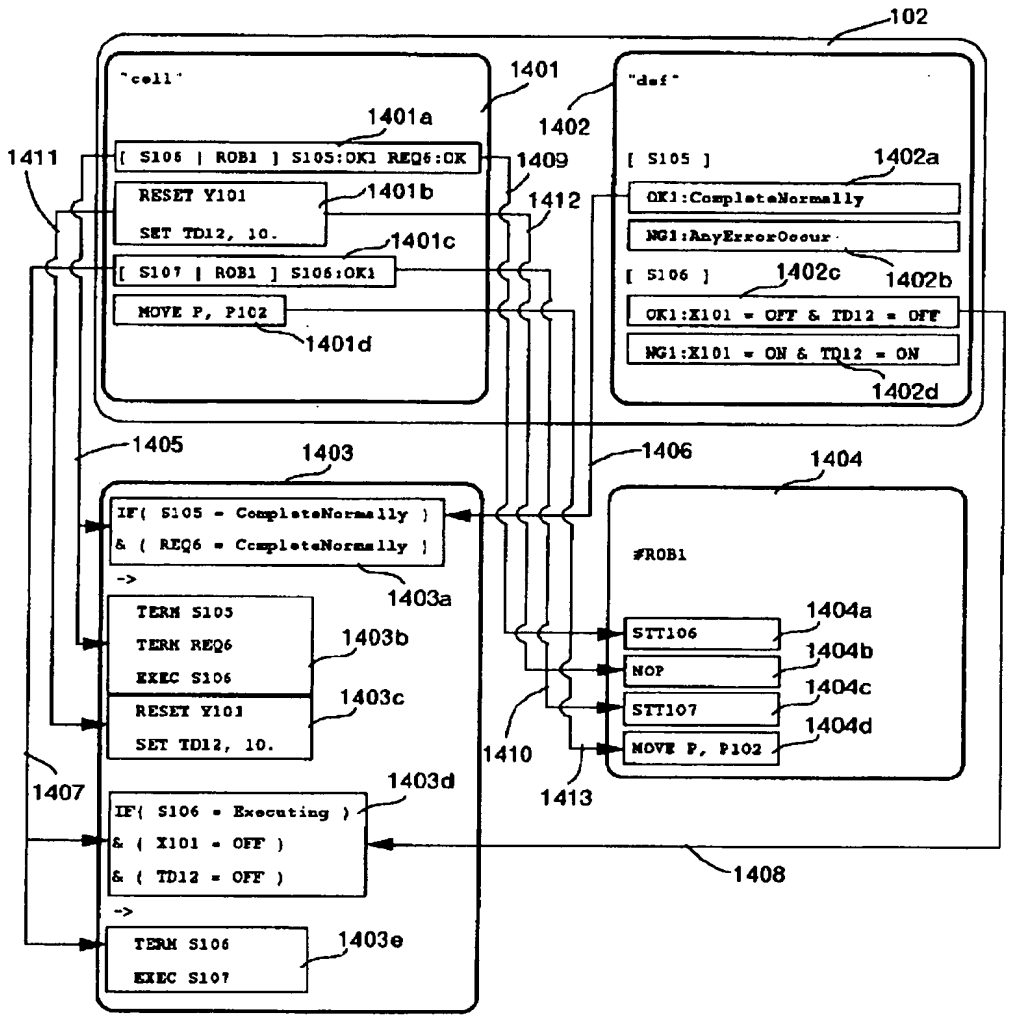
【図２３】


ロボット制御プログラム生成処理スタート
各ユニット毎のプログラムバッファ領域確保
1108a
ロボット動作タイミングデータから
状態NO.（=i）を獲得
1108b
状態ラベル"STTi"をバッファに出力
1108c
ロボット制御命令列
データなし？
1108d
YES
"NOP"を
バッファに出力
1108f
NO
ロボット制御命令列データの
内容をバッファに出力
1108e
次の状態あり？
1108g
YES
NO
次のユニットあり？
1108h
YES
NO
各ユニット毎のプログラムバッファを結合
1108i
完了

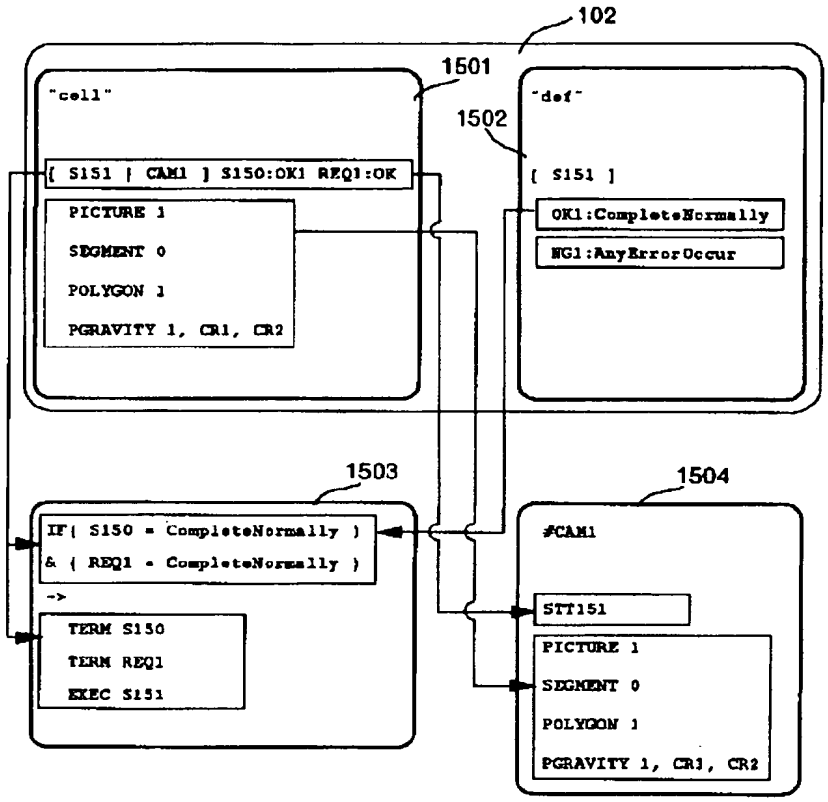
【図２４】


画像処理プログラム生成処理スタート
各ユニット毎のプログラムバッファ領域確保
画像処理タイミングデータから
状態NO.（=i）を獲得
状態ラベル"STTi"をバッファに出力
画像処理命令列データなし？
YES
"NOP"を
バッファに出力
NO
画像処理命令列データの内容を
バッファに出力
次の状態あり？
YES
NO
次のユニットあり？
YES
NO
各ユニット毎のプログラムバッファを結合
完了

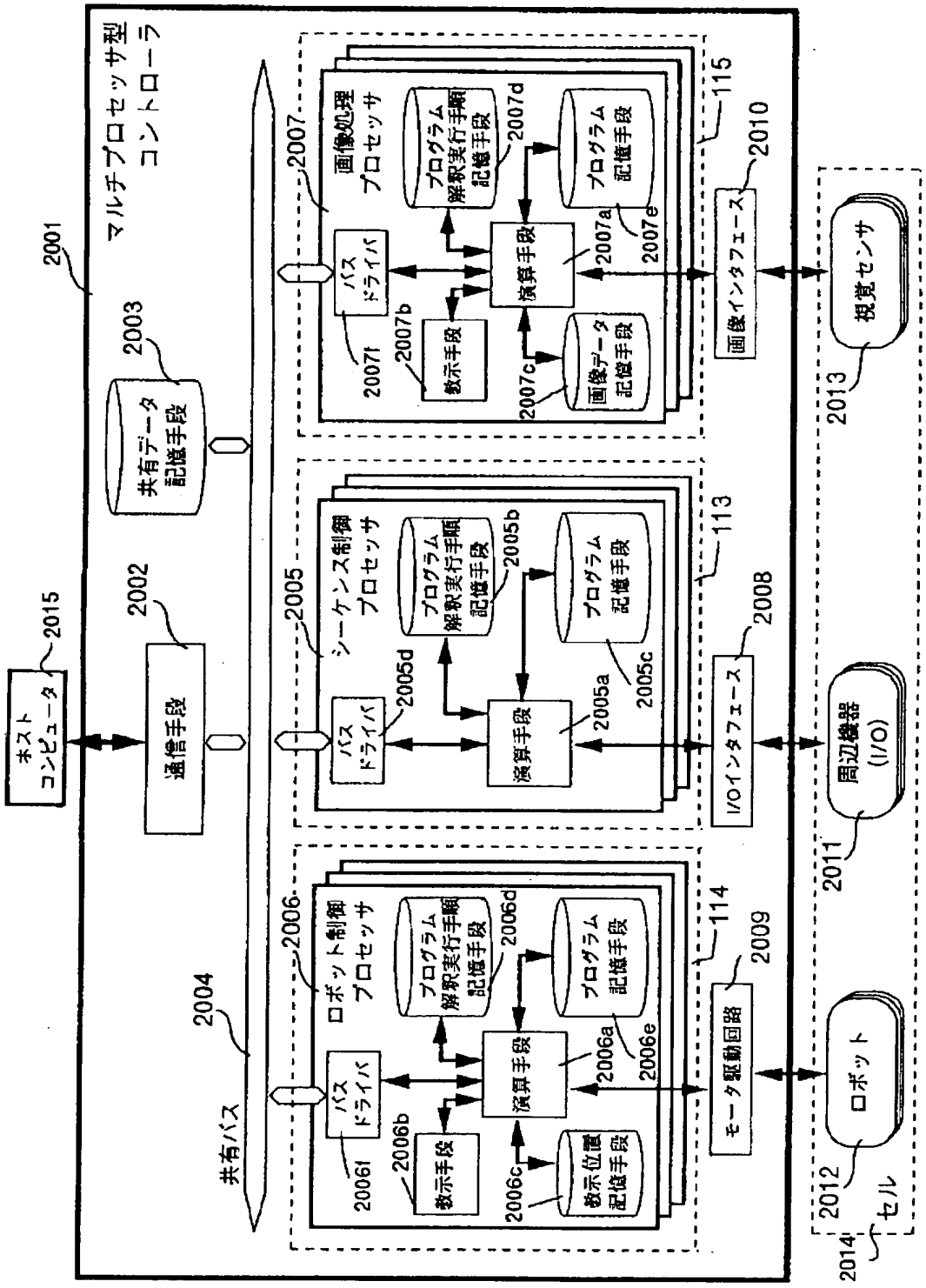
【図25】


102
"cell"
1401
1401a
[S105 | ROB1] S105:OK1 REQ6:OK
1411
1409
RESET Y101
1401b
SET TD12, 10.
1412
1401c
[S107 | ROB1] S106:OK1
MOVE P, P102
1401d
"def"
1402
[S105]
1402a
OK1:CompleteNormally
NG1:AnyErrorOccur
1402b
[S106]
1402c
OK1:X101 = OFF & TD12 = OFF
NG1:X101 = ON & TD12 = ON
1402d
1405
1403
1406
1404
IF(S105 = CompleteNormally)
& (REQ6 = CompleteNormally)
->
1403a
#ROB1
TERM S105
TERM REQ6
EXEC S106
1403b
1404a
STT106
1404b
NOP
RESET Y101
SET TD12, 10.
1403c
1404c
STT107
1404d
MOVE P, P102
1410
1413
1407
1403d
IF(S106 = Executing)
& (X101 = OFF)
& (TD12 = OFF)
->
1408
TERM S106
EXEC S107
1403e

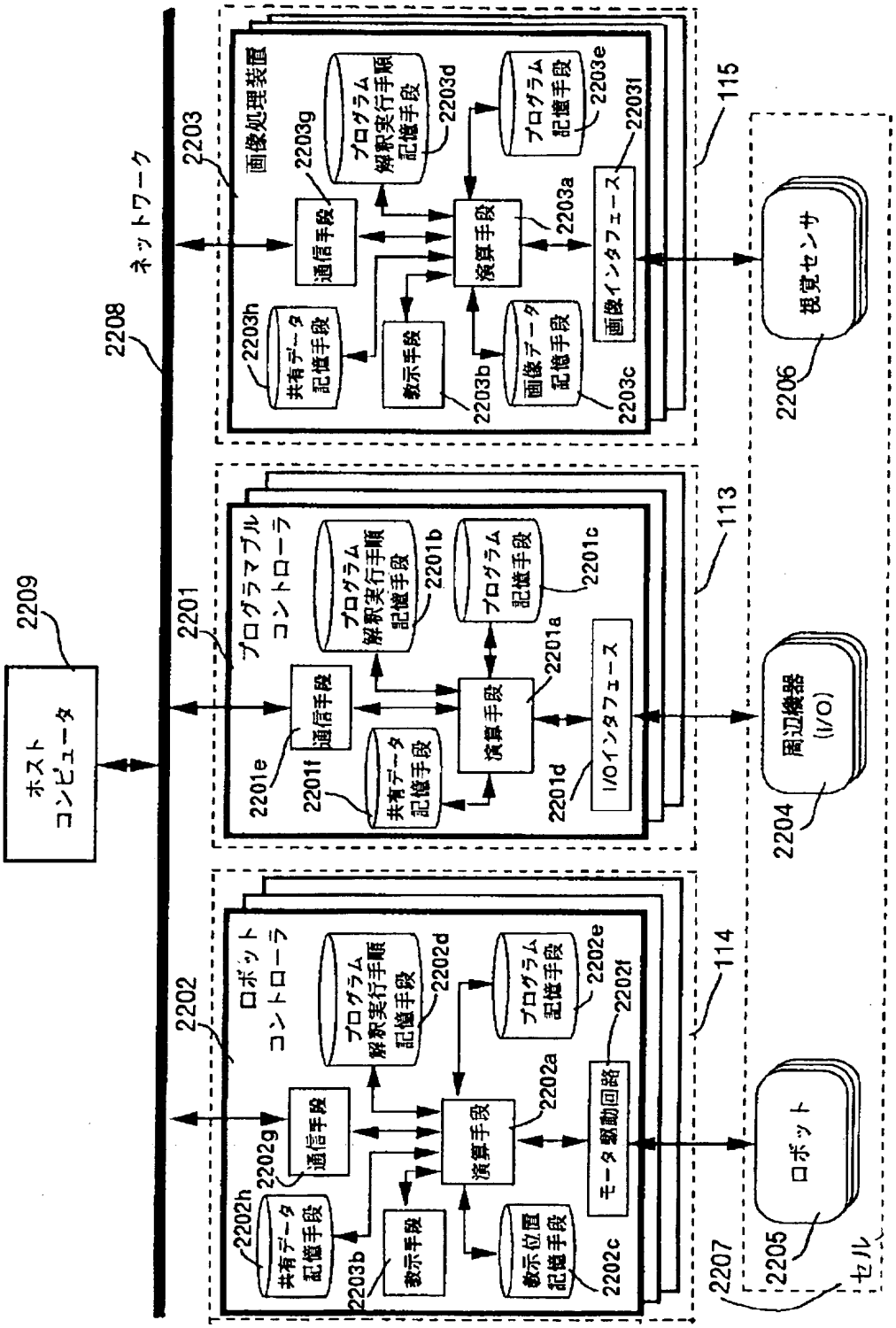
【図26】


102
1501
"cell"
{ S151 | CAM1 } S150:OK1 REQ1:OK
PICTURE 1
SEGMENT 0
POLYGON 1
PGRAVITY 1, CR1, CR2
1502
"def"
[S151]
OK1:CompleteNormally
NG1:AnyErrorOccur
1503
IF(S150 = CompleteNormally)
& (REQ1 = CompleteNormally)
->
TERM S150
TERM REQ1
EXEC S151
1504
#CAM1
STT151
PICTURE 1
SEGMENT 0
POLYGON 1
PGRAVITY 1, CR1, CR2

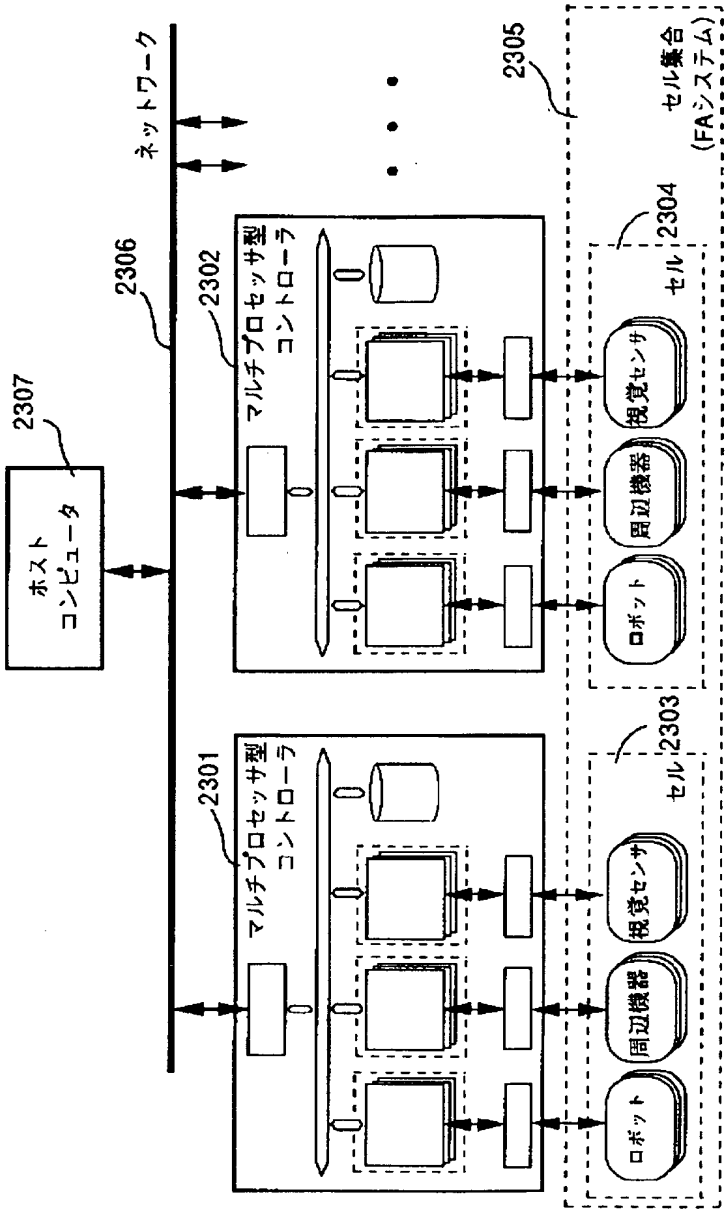
【図30】


マルチプロセッサ型
コントローラ
2001
ホスト
コンピュータ
2015
通信手段
2002
共有データ
記憶手段
2003
共有バス
2004
シーケンス制御
プロセッサ
2005
バス
ドライバ
2005d
プログラム
解釈実行手順
記憶手段
2005b
プログラム
記憶手段
2005c
演算手段
2005a
113
ロボット制御
プロセッサ
2006
バス
ドライバ
2006f
プログラム
解釈実行手順
記憶手段
2006d
プログラム
記憶手段
2006e
演算手段
2006a
表示手段
2006b
表示位置
記憶手段
2006c
114
画像処理
プロセッサ
2007
バス
ドライバ
2007f
プログラム
解釈実行手順
記憶手段
2007d
プログラム
記憶手段
2007e
演算手段
2007a
表示手段
2007b
画像データ
記憶手段
2007c
115
I/Oインタフェース
2008
モータ駆動回路
2009
画像インタフェース
2010
周辺機器
(I/O)
2011
視覚センサ
2013
ロボット
2012
セル
2014

【図32】


ホスト
コンピュータ
2209
ネットワーク
2208
2201
プログラマブル
コントローラ
2201e
通信手段
2201f
共有データ
記憶手段
2201d
I/Oインタフェース
2201a
演算手段
2201b
プログラム
解釈実行手順
記憶手段
2201c
プログラム
記憶手段
113
2202
ロボット
コントローラ
2202g
通信手段
2202h
共有データ
記憶手段
2203b
教示手段
2202a
演算手段
2202d
プログラム
解釈実行手順
記憶手段
2202c
教示位置
記憶手段
2202e
プログラム
記憶手段
2202f
モータ駆動回路
114
2203
画像処理装置
2203g
通信手段
2203h
共有データ
記憶手段
2203b
教示手段
2203a
演算手段
2203d
プログラム
解釈実行手順
記憶手段
2203e
プログラム
記憶手段
2203c
画像データ
記憶手段
2203i
画像インタフェース
115
2204
周辺機器
(I/O)
2206
視覚センサ
2205
ロボット
2207
セル

【図33】


ネットワーク
2305
セル集合
(FAシステム)
2306
2307
ホスト
コンピュータ
2302
マルチプロセッサ型
コントローラ
セル
2304
視覚センサ
周辺機器
ロボット
2301
マルチプロセッサ型
コントローラ
セル
2303
視覚センサ
周辺機器
ロボット